The Magnificent WORD PROCESSING SYSTEM

Model I

Version 2.1

Serial # AA121Ø113

Manual D

ANITEK T.M.
Software Products
P.O. Box 1136
Melbourne, FL 32935
Phone: 305-259-9397

LIMITED GUARANTEE

This product is sold on an "AS IS" basis and is without warranty. The advertisements used to promote this product and the manual provided with this product are believed to be fair descriptions of this product's capabilities, but are in no way guarantees of actual performance. If the purchaser discovers that this product will not meet his word processing needs or is in any other way dissatisfied with its performance, he may return it to place of purchase within thirty days from the date of purchase and receive a full refund. Returns must include the original sales slip, must be post-marked no later than thirty days after date of sale, and must contain the original manual and diskette in like-new condition with serial numbers on each intact.

WAIVER OF LIABILITY

ANITEK shall have no liability or responsibility to purchaser or any other person or entity regarding liability, loss, or damage caused, or alleged to be caused, directly or indirectly, by use of programs, program manuals, or segments of either which are part of this or any other product provided by ANITEK. This includes, but is not limited to, any loss of business or anticipated profits resulting from such loss of business, or any interruption of service or consequential damages, whether physical, monetary, material, or of any other kind, resulting from any use of such programs or program manuals, or segments of either.

DECLARATION OF COPYRIGHT

Welcome to the Magic of

Zorlof

The Magnificent

REQUIRED HARDWARE

1. TRS-80 model I or III.

2. 32K or more memory.

3. 1 or more disk drives

4. One of the following DOS's: TRSDOS, LODS, DOSPLUS, NEWDOS, NEWDOS/80, or MULTIDOS (single or double density).

5. Lower case modification (Model 1 only).

6. Line printer (can be either parallel or serial).

Note: Any standard line printer will work. However, using one which ZORLOF specially supports (see section 5.1) will give you more word processing power than using another brand.

RECOGNITION OF TRADEMARKS

1. TRS-80, TRSDOS, Line Printer IV, Line Printer VI, Daisy Wheel II, and Radio Shack are trademarks of Tandy Corporation.

2. Microline is a trademark of OKI Electric Industry Company, LTD.

3. Starwriter F-10, Starwriter FP-1500, and Prowriter 8510 are trademarks of C. Itoh Electronics, Inc.

4. Epson, MX-80, MX-100, Graftrax,and Graftrax Plus are trademarks of Epson America, Inc.

5. Centronics 737 and Centronics 739 are trademarks of Centronics Data Corporation.

6. Qume is a trademark of Qume Corporation.

7. Diablo 630 is a trademark of Diablo Systems, Inc.

8. Spinwriter 5510, 5515, 5520, 5525, and PC-8023A-C are trademarks of NEC America, Inc.

9. TEC 8500R is a trademark of Tec Corporation.

10. Brother HR-1 is a trade mark of Brother Industries, LTD.

11. ComRiter CR-1 is a trademark of Comrex International, Inc.

12. Smith-Corona TP-1 is a trademark of Smith-Corona.

13. LDOS is a trademark of Logical Systems, Inc.

14. NEWDOS and NEWDOS/80 are trademarks of Apparat, Inc.

15. DOSPLUS is a trademark of Micro Systems Software, Inc.

TABLE OF CONTENTS

Page

Page

APPENDIX

CHARACTER SYMBOLS

NOTE: The characters within parentheses throughout this manual are those character representations that are displayed on the Model I computer. These differ from those displayed on the Model III as follows:

CHARACTER	MODEL I	MODEL III
RETURN	←	]
SUPERSCRIPT	↑	[
SUBSCRIPT	↓	\
BLOCK MARKER	¥	~
BLANK	▒	±
SPECIAL FUNCTION	£	`
TAB	→	^
END-OF-SENTENCE	▬	▬

1. GETTING STARTED

1.1. Fill out and send in the enclosed registration card. This is our only way of knowing who has bought ZORLOF and how you can be reached. If a card was not provided with your system, please write your name, address, phone number, serial number of the disk, date of purchase, and a brief description of your hardware, and how you expect to be using ZORLOF on a piece of paper and mail to ANITEK, P. O. Box 1136, Melbourne, Florida, 32935.

1.2. Read the manual cover to cover. A thorough understanding of its contents is important. Then remove the the summary card from the back of this manual and refer to it as you use ZORLOF to help you remember what you read in the manual.

1.3. Copy ZORLOF and the example text files from the master disk to a TRSDOS compatible operating system disk. The file names are ZORLOF/CMD, SAMPLE/TXT, FORM/TXT, FORM/DAT. Once the original is copied, store it away in a safe place and don't ever use it for anything except making more copies.

1.4. Boot up your system with a DOS disk that contains ZORLOF/CMD. Type "ZORLOF", then hit the ENTER key. After a few seconds the screen will blank except for two status lines at the top of the screen (explained in Section 3) and a blinking rectangle at the start of the next line down (this is the cursor, the point at which your typing takes place). ZORLOF is now running.

1.5. Now you are ready to type. Be sure you have a pretty good understanding of the rest of the manual so you will know how everything works. The system was made very "user-friendly" so we're sure you will be taking command of ZORLOF's infinite powers in no time at all.

1.6. At this point you may wish to familiarize yourself with the sample text files supplied on the ZORLOF master disk. SAMPLE/TXT will demonstrate many of the PRINT FORMAT commands and how they work (explained in section 5). FORM/TXT and FORM/DAT together give an example of how a form letter text file and form letter data file might be set up (explained in section 7). Both of these examples can be run on your system if you first change the PRINTER SPECIFICATION commands (see 5.1) to that which matches your own printer.

1.7. Once you understand the manual and have tried some of the examples, we don't expect you to have any difficulties using ZORLOF. However, should you need any assistance or have a question concerning this product, just give us a call or drop us a note. We'd be more than happy to help you in any way we can.

2. WORD-WRAP AND JUSTIFICATION

The operator of ZORLOF is free to type continuously, never worrying about where to break the lines, or how the spacing between the words should be to achieve proper line justification. ZORLOF does this all for you automatically, using its advanced WORD-WRAP and LINE JUSTIFICATION capabilities which match and even surpass those found on many of the higher priced word processing systems. Both of these features will become very powerful text editing tools for you once they are understood.

2.1. WORD-WRAP

If a word that you are typing will not fit on the line without spilling over the end of the line (see 3.2 for line length specifications), then ZORLOF will automatically take that word off that line and put it on the next line. At the same time, the line you were typing on automatically justifies on the screen.

Another part of the WORD-WRAP capabilities of ZORLOF is REVERSE WORD-WRAP. Any time you type a space on a line, ZORLOF will check to see if the word(s) from the beginning of the line to that space can fit on the previous line. If it can then it is "wrapped backwards", so to speak, and put at the end of the previous line. The previous line is then re-justified and the cursor is moved back to the second column of the current line (second, because the space is in the first). ZORLOF will continue to move words up to the end of the previous line with each space you type until the previous line is full.

While scrolling up, the WORD-WRAP function in ZORLOF will always put as many words on a line as will fit. Lines containing a RETURN character (←)(4.47), however, are the exception. No text will ever be put on a line following a RETURN character while scrolling. At the same time, any text that you may have typed on a line following a RETURN character will be lost when that line is scrolled off the screen.

2.2. LINE JUSTIFICATION

ZORLOF has very advanced LINE JUSTIFICATION routines which automatically justify the lines of your text, not only as they are printed, giving you a very professional-looking copy, but also right on the screen as you type, giving you the opportunity to see how the text is going to look before you ever print it. You can format your text using one (or a combination) of four different LINE JUSTIFICATION modes. They are:

JUSTIFY LEFT — All text lines start in the first character column to form a smooth margin on the left side of the screen and paper.

JUSTIFY RIGHT — All text lines end in the last character column to form a smooth margin on the right side of the screen and paper.

JUSTIFY BOTH — All text lines start and end in the first and last character columns, respectively, to form smooth margins on both sides of the screen and paper. The spacing between words is done automatically.

JUSTIFY CENTER — All text lines are centered on screen and on paper.

If desired, you can change the LINE JUSTIFICATION mode at any place in your text -- having one paragraph set to JUSTIFY LEFT, the next to JUSTIFY CENTER, etc. The default justification mode is JUSTIFY LEFT. Refer to section 5.11 for instructions on how to set the LINE JUSTIFICATION mode within your text.

If you happen to own a printer type K3, K7, K8, K9, K11, K13, K14, K16, K20 (5.1), you will be pleased to know that ZORLOF can take advantage of the full power of the proportional-space text-printing feature that these printers have. Each line will be justified, as with any printer, but all the between-word spacings will be made exactly equal and the spaces between characters will be uniformly increased to avoid excessive between-word spacing, whenever necessary. The result is a printed page that looks almost like it was run off on an expensive typesetter.

Using ZORLOF with any one of these types of printers, you can also specify any character print density from 6.0 to 20.0 characters per line inch. The maximum print density you will actually be able to achieve and still have lines correctly justify will depend on the average character width for your printer. About 14 characters per inch is as tight as you will normally be able to print on a dot matrix printer, and about 11 characters per inch on a daisy wheel. This manual was printed in proportional-space mode at a character density of 11.0 characters per inch using NEC PC-8023A-C printer. Refer to section 5.16 for instructions on how to specify CHARACTER DENSITY within your text.

3. STATUS LINES

The top two display lines on your screen are reserved for status information. These lines can never be scrolled off, or removed from the screen.

When ZORLOF first comes up, the cursor will be positioned over to the left just under the second STATUS LINE. The only way to get the cursor into the STATUS LINES from this point (or from any point in the text portion of the display) is to hold down the CLEAR key and strike the "=" key. This is denoted CLEAR-= (see 4.50).

You will notice that ZORLOF will only allow the cursor to go into certain areas of the STATUS LINES and not others. This is a protective measure to keep you from typing over any part of the STATUS LINES that should not be changed. Also, some functions are not operable while the cursor is in the STATUS LINE (see begining of section 4 for list). This, too, is precautionary to protect the STATUS LINES.

There are 5 fields in the first STATUS LINE. Two of them, the NAME FIELD and the WIDTH FIELD, are for you to type information in to tell ZORLOF certain things about the text you are working on. The other three, the WORD COUNT FIELD, the LINE COUNT FIELD, and the FREE MEMORY FIELD are for ZORLOF to write information in to tell you certain things about the text. The second STATUS LINE contains the SEARCH FIELD and the REPLACE FIELD, which you use to type the SEARCH and REPLACE strings in while using those respective functions (4.37, 4.38). When needed, ZORLOF will use the second STATUS LINE for displaying the ERROR MESSAGES. Each of these fields is discussed in detail on the following pages.

3.1. NAME FIELD

This is the field where you type the name of the text file. This is the name that ZORLOF will use when you go to file your text onto the disk. The name must follow the file specification format as dictated by your disk operating system (consult your DOS operator's manual), and must be no longer than 14 characters, including the extension and drive specification (if used). Passwords are not allowed as part of the file name.

Also, anytime you get a text file from the disk, unless there is already text in the computer's memory at the time, ZORLOF will take the name that you have used in specifying that file and write it into the **NAME FIELD** for you. This saves you some trouble and it also helps to prevent you from writing the file back out to disk under the wrong name.

3.2. WIDTH FIELD

This is the field where you type the maximum width (in characters) of your copy. This is also referred to as maximum line length in other parts of this manual. Any width from 5 to 128 may be entered in this field. Portions of text can be made narrower than this maximum specification by implementing the **LEFT AND RIGHT SIDE TEXT INDENTING** printer commands as described in section 5.12.

Since your display screen is limited to only 64 character lines, ZORLOF will automatically assign two display lines for each text line any time you specify text lines to be greater than 64 characters. In this case, the **WORD-WRAP** and **JUSTIFICATION** routines will treat each second line as an extension of the line before it and not as a separate line. You will get a better feel for how this is done when you begin to use ZORLOF.

It is possible to type your text in one width and then change it later to a different width if you so desire. Just type the new width in the WIDTH FIELD and hit CLEAR-T (4.39). Voila, the text is automatically reformatted, word-wrapped, and line justified to the new line length.

You may also be happy to know that the information in the WIDTH FIELD is saved as part of the file data every time you file the text (or even a portion of the text -- see 4.22) onto disk. Then, the next time you call that text file back to the screen to work on it, ZORLOF will automatically extract that information from the file and write it into the WIDTH FIELD for you, saving you a lot of time and trouble. If, however, you are getting a text file from disk to insert it into text that is already on the display screen, the width information of that second file is just stripped from the text data of the file and ignored, maintaining the width of the first file.

The WIDTH FIELD is used in a slightly different way when editing files other than ZORLOF-type text files. During these times, the width field will contain a three character abreviation for the type of file being edited. If you are using ZORLOF to create one of these non-ZORLOF-standard files you will need to type the appropriate 3 characters in the WIDTH FIELD before you start. If you are using ZOROF to edit a pre-existing file, then the correct three character abreviation will appear in the WIDTH FIELD automatically as the file is being loaded in from the disk. The various types of non-standard files that ZORLOF can create and/or edit, along with their three character abreviations, and explanations of the unique characteristics of each type of file are given in Appendix A, in the back of this manual. I you have any desire to use ZORLOF for editing BASIC programs, Assembly language source files, other word processing systems' text files, or patching machine code programs, it is recomended that you review Appendix A.

3.3. WORD COUNT FIELD

This field contains a dynamic count of the total number of words in your text file, both on the screen and in memory, and is updated each time you strike a key. A word is defined by a character or a string of characters separated from other characters by one or more SPACE characters ( ), BLANK characters (▒), END-OF-SENTENCE characters (▬), TAB characters (→), UNDERLINED SPACE characters (_), the start of a line, or the end of a line. Characters in printer command lines (lines starting with PRINTER COMMAND characters; see 4.16) are not counted. The information contained in this field is maintained by ZORLOF and cannot be changed manually by the user.

Note: During ZAP-PROCESSING (4.58) the data contained in this field will be irrelevant and will not correspond to the number of words in the file.

3.4. LINE COUNT FIELD

This field contains a dynamic count of the total number of lines in your text file, both on the screen and in memory, this is updated each time you strike a key. A text line is equal to one display line if your text width is 64 characters or smaller, or equal to two display lines if your text width is 65 characters or larger. Printer command lines (lines starting with PRINTER COMMAND characters, see 4.16) are not counted. The information contained in this field is maintained by ZORLOF and cannot be changed manually by the user.

Note: During ZAP-PROCESSING (4.58) the data contained in this field will be irrelevant and will not correspond to the number of lines in the file.

3.5. FREE MEMORY COUNT FIELD

This field contains a dynamic count of the unused memory space still available for character entry, and is updated each time you strike a key. The information contained in this field is maintained by ZORLOF and cannot be changed manually by the user.

Note: During ZAP-PROCESSING (4.58) the data contained in this field will be irrelevant and will not correspond to the actual free memory.

3.6. SEARCH FIELD

If there is some string of characters that you want to SEARCH through your text for, you can have ZORLOF do it for you automatically. Type the string of characters in the SEARCH FIELD (maximum 28 characters) and hit CLEAR-S (4.37). ZORLOF will automatically scan the entire text, from the cursor down, looking for that string of characters. If found, the cursor will stop on the first character of the string. If not found, the cursor will stop on the next empty line after the end of the text.

When entering the character string in the SEARCH FIELD you can use upper case or lower; it really doesn't matter. ZORLOF will look for that string in any combination of upper case or lower case.

The BLANK character (⋇) in the SEARCH string is used as a wild card. Use it when you wish to SEARCH for a string in which some of the characters in the string have to be certain things, but other characters can be anything.

Use the RETURN character (←) in the SEARCH FIELD to define the end of a SEARCH string, otherwise it is assumed to be the right-most non-space character in the field. This is necessary if you want to search for the word "at", but you don't want the cursor to stop on the word "attack". In this case the SEARCH string should be entered: space, "a", "t", space, RETURN character. The space after the "at" is now taken to be the last character of the string, and ZORLOF will know just to stop on the word "at", and not "atom" or "cat".

3.5. FREE MEMORY COUNT FIELD

This field contains a dynamic count of the unused memory space still available for character entry, and is updated each time you strike a key. The information contained in this field is maintained by ZORLOF and cannot be changed manually by the user.

Note: During ZAP-PROCESSING (4.58) the data contained in this field will be irrelevant and will not correspond to the actual free memory.

3.6. SEARCH FIELD

If there is some string of characters that you want to SEARCH through your text for, you can have ZORLOF do it for you automatically. Type the string of characters in the SEARCH FIELD (maximum 28 characters) and hit CLEAR-S (4.37). ZORLOF will automatically scan the entire text, from the cursor down, looking for that string of characters. If found, the cursor will stop on the first character of the string. If not found, the cursor will stop on the next empty line after the end of the text.

When entering the character string in the SEARCH FIELD you can use upper case or lower; it really doesn't matter. ZORLOF will look for that string in any combination of upper case or lower case.

The BLANK character (▒) in the SEARCH string is used as a wild card. Use it when you wish to SEARCH for a string in which some of the characters in the string have to be certain things, but other characters can be anything.

Use the RETURN character (←) in the SEARCH FIELD to define the end of a SEARCH string, otherwise it is assumed to be the right-most non-space character in the field. This is necessary if you want to search for the word "at", but you don't want the cursor to stop on the word "attack". In this case the SEARCH string should be entered: space, "a", "t", space, RETURN character. The space after the "at" is now taken to be the last character of the string, and ZORLOF will know just to stop on the word "at", and not "atom" or "cat".

3.7. REPLACE FIELD

Sometimes when you SEARCH for a string of characters you will want to REPLACE it with another string of characters. You can do so using this field and the REPLACE function (4.38).

Type the replacement string of characters in the REPLACE FIELD just as you would want it to appear in your text. BLANK characters (▓) count as actual BLANK characters. The case of each character (upper or lower) does matter, so type each character in the case that you want it. The RETURN character (←) can be used in the REPLACE FIELD the same way it is used in the SEARCH FIELD if you want the REPLACE string to end with a space instead of a character.

Once you have typed the replacement string in the REPLACE FIELD, hit CLEAR-S to SEARCH for the text string that you are going to REPLACE, then hit CLEAR-R. The string of characters that you have SEARCHED for will be removed at the cursor, and a copy of the string of characters in the REPLACE FIELD will be automatically written in its place.

The REPLACE string, by definition, is a replacement for the SEARCH string. Thus, hitting CLEAR-R at any time that the cursor is not positioned at the start of a portion of text that matches the SEARCH string, nothing will happen.

While we are on the subject, you might as well be introduced to one more capability of ZORLOF which relates to the SEARCH and REPLACE functions -- the AUTOMATIC SEARCH AND REPLACE function. This function will automatically SEARCH through your entire text, from the cursor down, and change every occurrence of the SEARCH string with a copy of the REPLACE string. See section 4.19 for more information on the AUTOMATIC SEARCH AND REPLACE function.

Note: AUTOMATIC SEARCH AND DELETE is done the same way as the AUTOMATC SEARCH AND REPLACE except that the REPLACE FIELD is left blank.

3.8. ERROR MESSAGE FIELD

You cannot see this field when ZORLOF first comes up, but it really is there. Actually, under certain error conditions, the SEARCH FIELD and the REPLACE FIELD are temporarily removed and stored in the computer's memory. This leaves the entire second line available for the ERROR MESSAGE. The appropriate ERROR MESSAGE flashes on the screen a few times, letting you know what has happened. Then the SEARCH FIELD and REPLACE FIELD return to the screen with the same information they had before the error condition occurred, and you are free to take whatever measures are necessary to correct the error condition, such as delete an obsolete file if your disk is full, etc.

The error conditions that ZORLOF checks for and the ERROR MESSAGES that result are as follows:

"DISK ERROR, TRY AGAIN"
"DISK FULL"
"WRITE PROTECTED DISKETTE"
"DISK NOT READY"
"FILE PROTECTED"
"NO SUCH FILE"
"DIRECTORY FULL"
"MEMORY FULL"
"BAD BLOCK"
"QUEUE IS FULL"
"ONCE MORE TO CLEAR"
"ONCE MORE TO DELETE"
"ONCE MORE TO KILL"

Most of these are self explanatory except for maybe the last five. "BAD BLOCK" refers to an illegal placement of the BLOCK MARKER characters while using the COPY BLOCK, MOVE BLOCK, or STORE BLOCK functions (4.20, 4.21, 4.32, 4.33). "QUEUE IS FULL" means that you have tried to enter more than 13 file names in the PRINT QUEUE (4.36). "ONCE MORE TO CLEAR" is the warning message on the double acting RESET function (4.53). "ONCE MORE TO DELETE" is the warning message for the double acting DELETE BLOCK function (4.46). "ONCE MORE TO KILL" is the warning message on the double acting KILL FILE function (4.29).

4. EDITING FUNCTIONS

Editing functions are those which assist the operator in preparing and manipulating text while it's still on the screen and in the computer's memory. These are functions such as: moving a block of text from one place to another, deleting, searching for a particular word(s), scrolling, or calling up a disk directory, just to name a few.

The following pages contain summaries of all 61 editing functions available to the ZORLOF user, how each is invoked, and an explanation of how each works. Most functions are invoked by holding down the CLEAR key and then striking one of the other keys. The SEARCH function is one example. It is denoted in this section like:

CLEAR-S

Others are invoked by holding the SHIFT key down and striking one of the other keys. For example, SCROLL UP will be denoted:

SHIFT-↑

And still others like CURSOR RIGHT are invoked by a single key:

→

Note: The following functions are not operative while the cursor is in either one of the STATUS LINES:

4.11	DELETE CHARACTER
4.12	DELETE CHARACTER BACKWARDS
4.21	COPY BLOCK
4.25	GET TEXT FROM DISK
4.30	TAB
4.31	DELETE LINE
4.32	MOVE BLOCK
4.34	OPEN LINE
4.42	DELETE WORD
4.44	SPLIT TEXT
4.46	DELETE BLOCK

4.1. CURSOR RIGHT →

The cursor is moved one column to the right. After the cursor has reached the last column of the line the cursor will then skip down to the first column of the next line.

4.2. CURSOR LEFT ←

The cursor is moved one column to the left. No action is taken if the cursor is in the first column of the line.

4.3. CURSOR UP ↑

The cursor is moved one line up. If the cursor is on the top line of the display the text will be scrolled down instead.

4.4. CURSOR DOWN ↓

The cursor is moved one line down. If the cursor is on the bottom text line of the display the text will be scrolled up instead.

4.5. CURSOR FAR RIGHT SHIFT-→

The cursor is moved to one column past the last character of the line. This function can also be used to place the cursor in the right most field of the STATUS LINES.

4.6. CURSOR FAR LEFT SHIFT-←

The cursor is moved to the first column of the line. This function can also be used to place the cursor in the left most field of the STATUS LINES.

4.7. SCROLL UP SHIFT-↑

The text is scrolled up one line.

4.8. SCROLL DOWN SHIFT-↓

The text is scrolled down one line.

4.9. PAGE UP CLEAR-↑

The text is scrolled up thirteen lines if the maximum text width as set in the WIDTH FIELD is 64 characters or less, 6 lines if greater than 64 (see 3.2).

4.10. PAGE DOWN CLEAR-↓

Same as 4.9 except down instead of up.

4.11. DELETE CHARACTER CLEAR-→

The character at the cursor is deleted and the text to the right of the cursor is shifted left one column to fill the gap. If the cursor is to the right of the last character on the line then the cursor will move to the first column of the next line. If the cursor is in the first column of a blank line then all the text below the cursor will scroll up one line to fill the gap.

Note: When the DELETE CHARACTER function is used on an expanded character, both the character and the vertical line in front of it (¦)(used to indicate that it's an expanded character, see 4.43) are deleted.

4.12. DELETE CHARACTER BACKWARDS CLEAR-←

Same as 4.11 except that the cursor is moved one column to the left first. No action is taken if the cursor is in the first column.

4.13. LOWER CASE CLEAR-.

An alpha character (A-Z) at cursor is changed to lower case (if not already) and the cursor is moved one column to the right.

4.14. UPPER CASE CLEAR-,

An alpha character (A-Z) at cursor is changed to upper case (if not already) and the cursor is moved one column to the right.

4.15. CAPITAL LOCK CLEAR-/

All alpha characters (A-Z) entered after hitting CLEAR-ENTER will be in upper case. Hit CLEAR-ENTER a second time to terminate.

4.16. PRINTER COMMAND CHARACTER CLEAR-;

A PRINTER COMMAND character (<) is entered at the cursor. All characters on the same line and following this character will be assumed by the printing routine in ZORLOF to be printer commands. Also, all lines starting with this character will be stripped from the text while PRINTING (4.35) or VIEWING (4.41).

Note: If the PRINTER COMMAND character is typed at some place other than the first character of a line, the WORD-WRAP and LINE JUSTIFICATION routines in ZORLOF will automatically put it, and the characters which follow, on their own line as soon as that text is scrolled off and then scrolled-up back onto the screen, or when CLEAR-J is struck (4.28).

4.17. SPECIAL FUNCTION CHARACTER SHIFT-@

The SPECIAL FUNCTION character (£) is entered at the cursor. This character is used in conjunction with the SPECIAL FUNCTION printer command to "define" and "locate" special characters and printer functions within a line of text. This is explained in detail in section 5.27.

4.18. PRINT SCREEN CLEAR-@

The contents of the display screen is printed. This can be done while text is being edited, while text is being VIEWED (4.41), or while a DIRECTORY is on the screen (4.22). The STATUS LINES will not be printed in the case of the DIRECTORY. The type of printer assumed (parallel, serial, baud rate, etc.) will be that which was last specified when printing a text file (see 5.1, 5.2, and 5.3). If there was none previously specified, then the default is the parallel line printer. The function will terminate automatically when screen is fully printed, or manually by hitting CLEAR-Z.

4.19. AUTOMATIC SEARCH AND REPLACE CLEAR-A

All occurrences of the SEARCH string in the text, from the cursor down, are replaced by the REPLACE string. Terminate the function by hitting CLEAR-Z. See 3.6 and 3.7 for more information on how the SEARCH and REPLACE functions work.

4.20. BLOCK MARKER INSERT CLEAR-B

A BLOCK MARKER character (¥) is inserted into the text just ahead of the cursor. Two such BLOCK MARKER characters must be used to define a block of text, one for its beginning and one for its end, before the COPY BLOCK (4.21), MOVE BLOCK (4.32), or STORE BLOCK (4.33) functions can be implemented. Only one BLOCK MARKER character is needed when implementing the DELETE BLOCK function (4.46).

4.21. COPY BLOCK CLEAR-C

Once a block of text has been defined (4.20) that block can be copied to any place in your text simply by moving the to that place and hitting CLEAR-C. The original block will remain where it was and the block markers will be removed.

4.22. DIRECTORY CLEAR-D

At any time, whether text is on the screen or not, you can get a DIRECTORY of any disk by hitting CLEAR-D. The DIRECTORY will give you a complete list of files found on any disk on your system, the length of each file in bytes and granules, the amount of free space of each disk in granules, and the PRINT QUEUE list (4.36). Upon hitting CLEAR-D the first directory to be displayed is the one for disk drive 0. The directory for any other drive can be examined just by hitting 0, 1, 2, 3, etc. on the keyboard for the desired drive.

Only 13 file names can be displayed on the screen at one time. By hitting the space bar on your keyboard, the next set of 13 from disk can be examined. To exit from the DIRECTORY function, hit CLEAR-Z and your text will return to the screen.

Note: The same PRINT QUEUE list appears on the right side of the screen no matter what directory page is displayed on the left side of the screen.

4.23. END OF TEXT CLEAR-E

The text is scrolled up until the last line appears at the bottom of the screen. The cursor is then moved to the first blank line after the last line of text.

4.24. FILE TEXT TO DISK CLEAR-F

The text which is in the computer's memory and being displayed on the screen will be filed onto the disk according to the file name as it appears in the name field of the STATUS LINE (3.1). To file the text to disk as an "ASCII" file (non-ZORLOF-standard) the three characters "ASC" must be typed in the WIDTH FIELD prior to hitting CLEAR-F. See Appendix A for details on the differences of these two file types. Once filed, the text will appear back on the screen so you can resume working on it. If you would like the display and memory to clear after the text is filed, this can be done by holding down both the SHIFT key and the CLEAR key while hitting the "F" key.

4.25. GET TEXT FROM DISK CLEAR-G

There are two ways to invoke this function. You may call a DIRECTORY to the screen (4.22), move the cursor to the desired file name, and hit CLEAR-G. If there is text already in the computer's memory, then the new text will be inserted at the place where your cursor was just before you enacted the DIRECTORY function. If you get a second or third file at this time, they will be inserted following the ones before it. If there was no text in the computer's memory at the time, then the name and width of the first file will automatically be placed in their respective fields in the STATUS LINE (3.1, 3.2).

The second way to invoke this function is to bypass the DIRECTORY and just type the name of the file on a blank line (see 4.34 to open a blank line) and hit CLEAR-G. The line containing the file name will be removed from the text and the file with that name will be read into the computer's memory just as it did in the DIRECTORY example.

Note: File names must comply with the rules set forth in your DOS operator's manual, must be left justified on the line in which they were typed, must be no longer than 14 characters including the extension and drive specification (if used), and must not contain paswords.

4.26. HOME CURSOR — CLEAR-H

The cursor is moved to the first column of the top display line.

4.27. INSERT — CLEAR-I

The cursor will blink at a much slower rate to let you know that the INSERT function is active. As you type, all characters to the right of the cursor will be pushed right as characters are inserted at the cursor. As words reach the end of the cursored line they are automatically removed and put on the following line. The INSERT function can be terminated by hitting CLEAR-I a second time.

4.28. REJUSTIFY — CLEAR-J

As you type you will notice that ZORLOF does a marvelous job of keeping all the text lines justified except for the one that the cursor is on. This was done intentionally so that the words on the line of text where you are typing would stay still, and not be jumping around every time you hit a key.

This function, however, can be used to totally REJUSTIFY the section of text that is on the screen if you wish to see it in its fully justified form before moving on to another section of text. After hitting CLEAR-J, the text is removed from the screen and returned fully WORD-WRAPPED and LINE JUSTIFIED.

4.29. KILL FILE ON DISK CLEAR-K

Although this command (CLEAR-K) is invoked in the same way as the TAB function (4.30), it's doubtful that you will get the two confused. The TAB function only works when there is text on the screen, and the KILL FILE function only works when a DIRECTORY is on the screen.

To implement the KILL FILE function, just call the DIRECTORY to the screen (4.22), position the cursor over the file you wish to eliminate from the disk, and hit CLEAR-K. The first time CLEAR-K is struck the "ONCE MORE TO CLEAR" message will flash on and off in the second STATUS LINE. If you change your mind at this point just hit any other key to abort. Hitting CLEAR-K a second time will remove the file from the disk. The DIRECTORY is then recalled, showing you that it is no longer listed in the directory.

This function is especially useful if you need to file away to disk the text you are working on but the diskette is already full. You can KILL something on the disk to make room for the new text file, without ever losing the text in the computer's memory.

4.30. TAB

CLEAR-K

The cursor is moved to the next TAB STOP position and TAB characters (→) will be written from the point where the cursor was, up to the next TAB STOP. See section 5.4 for the setting of TAB STOPS.

TAB characters function very much like the BLANK characters (▒)(4.48) when being PRINTED (4.35) or VIEWED (4.41): both create extra spacing between words. However two differences do exist. Each set of TAB characters (from one TAB STOP to the next) is changed into a single TAB ASCII CODE (09H) when the file is written to disk. Likewise, when reading a text file from disk, all TAB ASCII CODES are automatically replaced by the appropriate number of TAB characters. The BLANK character, however, is always a BLANK character, whether on the screen, in memory, or on the disk.

The other difference is in how the two are treated when printing in proportional-space mode. In mono-space mode the rule is simple, all characters are the same width and each TAB character and each BLANK character are printed as mono-space SPACE characters -- no problem. But, on a proportional-space printer, the distance from the last character before a TAB to the first character after a TAB can vary considerably from one line to the next, since the characters are not all the same width. ZORLOF precisely calculates this distance for each TAB in each line to insure that all the TAB STOPS line up straight down the page. The printed width of the BLANK character, on the other hand, is set at some fixed value, which is inversely proportional to the value you set for the CHARACTER DENSITY (5.16), and does not change from line to line.

When using the TAB function, it is recommended that you do so only on lines that are set to JUSTIFY LEFT or JUSTIFY BOTH (2.2, 5.11). The meaning of a TAB in a line set to JUSTIFY RIGHT or JUSTIFY CENTER is obscure and the results would be unpredictable.

For the use of this function while editing EDTASM or BASIC files see section 6.4

Note: If this function is used while the DIRECTORY is on the screen, it will be interpreted as the KILL FILE command (4.29) instead of the TAB command.

4.31. DELETE LINE CLEAR-L

All the text from the cursor to the end of the line will be deleted. If the cursor is in the first column then the whole line will be deleted and the lines under the cursor will scroll up to fill the gap.

4.32. MOVE BLOCK CLEAR-M

Once a block of text is defined (4.20) that block can be moved to any place in your text by simply moving your cursor to that place and hitting CLEAR-M. Once the block is moved the block markers are removed.

4.33. STORE FILE TO DISK CLEAR-N

Once a block has been defined (4.20), that block can be written onto disk using this function. On a blank line (see 4.34 to open a blank line), write the desired name you wish that block to be filed under, then hit CLEAR-N. A copy of that block will be written to disk under that name, and the BLOCK MARKER characters, as well as the block's file name, will then be removed from the text.

Note: File names must comply with the rules set forth in your DOS operator's manual, must be left justified on the line in which they were typed, must be no longer than 14 characters including the extension and drive specification (if used), and must not contain passwords.

4.34. OPEN LINE CLEAR-O

All text lines below the cursor are scrolled down to open one blank line. The cursor is then moved to the first column of the blank line.

4.35. PRINT TEXT CLEAR-P

The text file will be sent to the printer in its final letter-quality form. See section 5 for details on printer initializing and print formatting. Once the text file that is in the computer's memory has been completely PRINTED, the function is terminated and the top portion of the text appears back on the screen. The only exception to this is when there is one or more files in the PRINT QUEUE (4.36). In this case, after the first file is PRINTED, the memory is cleared, the next file in the PRINT QUEUE is read from disk into memory, and it is PRINTED. The function will automatically terminate when the last file in the PRINT QUEUE is PRINTED, or manually by hitting CLEAR-Z.

4.36. QUEUE FILES FOR PRINTING CLEAR-Q

Up to 13 files can be registered in the PRINT QUEUE by calling the DIRECTORY to the screen (4.22), positioning the cursor over the desired file name, and hitting CLEAR-Q. The names of the files in the PRINT QUEUE will then be entered in First-In-First-Out order as seen on the right side of the screen. Entries may be removed from the PRINT QUEUE by positioning the cursor over the file name in the PRINT QUEUE and hitting CLEAR-L.

4.37. SEARCH CLEAR-S

The text will be SEARCHED from the positon after the cursor down to find a string of characters that matches the string of characters you have typed in the SEARCH FIELD (3.6). If found, the cursor will stop at the first character of that string. If not found, the cursor will stop on the next blank line after the last line of your text.

4.38. REPLACE CLEAR-R

If the cursor is positioned at the start of a string of characters that matches the string of characters you have typed in the SEARCH FIELD (3.6), those characters, upon hitting CLEAR-R, will be REPLACED with a copy of the string of characters which you have typed in the REPLACE FIELD (3.7). If the characters in the SEARCH FIELD do not match those starting at the cursor position, nothing will happen.

4.39. TOP OF TEXT CLEAR-T

The portion of your text which is currently on the screen is scrolled off the screen and the top portion of the text is scrolled onto the screen.

4.40. UNDERLINE CLEAR-U

When CLEAR-U is struck the character at the cursor is changed into a graphics character by the setting of the most significant bit of the 8 bit value that represents that character in the computer's memory. All such graphics characters will later be printed as their original character with an underline. The underline can be removed by putting the cursor back over the character and hitting CLEAR-U again.

There are four different ways to create underlines with ZORLOF, this is only one of them. It is recomended that you review all four methods and then use the method(s) that you feel most comfortable with. The other three methods are described in sections 4.57, 5.18, and 5.27.

4.41. VIEW TEXT CLEAR-V

Implementing this function will allow the user to VIEW his text in final format with proper indents, page breaks, header lines, footer lines, page number lines, and page numbers installed, non-printing characters (END-OF-SENTENCE characters, TABS, BLANKS, and RETURNS) replaced by SPACES, printer command lines removed, and all text fully justified -- all without ever printing it on paper. Because of the limitations of the TRS-80 hardware, however, certain things like depth of margin, bold, underline, superscript and subscript cannot be displayed on the screen during VIEWING and must be printed to be seen.

Hitting the DOWN ARROW key (↓) will cause the text to scroll forward one line. Holding this key down will cause continuous scrolling. If desired, you can scroll to a certain place in your text, then hit CLEAR-P, and all the rest of your text, from the last line displayed on your screen through to the end, will be printed on your printer. Also, if desired, you may get a print-out of just what is on the screen at any time by hitting CLEAR-@ (4.18). Terminating the VIEW function is done by hitting CLEAR-Z.

Note: It is not possible to scroll backwards through the text while implementing the VIEW function.

4.42. DELETE WORD CLEAR-W

All characters from the nearest space to the left of the cursor through the nearest space to the right of the cursor are deleted. All characters on the cursored line to the right of those deleted are then shifted left to fill in the gap. If the cursor is on a SPACE while trying to implement the DELETE WORD function, nothing will happen.

4.43. EXPANDED CHARACTERS CLEAR-X

When the EXPANDED CHARACTER function is active, all characters typed will appear on the screen as a thin disjointed line (¦) followed by the character you type. All text entered this way will get printed on your printer as an expanded character, provided your printer has that capability. Of the printers that ZORLOF supports, the printer types that have the capability of printing EXPANDED characters are: K1, K2, K3, K4, K5, K6, K7, K8, K9, K19, and K20 (5.1). If your printer is not one of these then it will be printed as a space followed by the normal-width character. Terminating the EXPANDED CHARACTER function is done by hitting CLEAR-X a second time.

Note: The following characters will be displayed on the screen as single-width characters even if the EXPANDED CHARACTER function is active: BLANK characters (▒), END-OF-SENTENCE characters (▬), TAB characters (→), PRINTER COMMAND characters (<), RETURN characters (←), SUPERSCRIPT characters (↑), SUBSCRIPT characters (↓), and SPACES ( ).

4.44. SPLIT TEXT CLEAR-Y

If you have a need to insert type at a certain place in your text, but you want to do it with a blank screen, then this is the function you want to use. Move the cursor to the place you want to insert the text and hit CLEAR-Y. All text before the cursor scrolls up and off the screen, and all text at and following the cursor scrolls down and off the screen.

Even though this is not a function that you have to terminate, you will have to do something to get your disjointed segments of text into one piece again. This can be done by either scrolling down an excess of 14 lines then scrolling up the same distance, or by hitting CLEAR-J (4.28).

4.45. TERMINATE FUNCTION CLEAR-Z

The following functions are terminated by hitting CLEAR-Z:

4.18	PRINT SCREEN	(CLEAR-@)
4.19	AUTOMATIC SEARCH AND REPLACE	(CLEAR-A)
4.22	DIRECTORY	(CLEAR-D)
4.35	PRINT TEXT	(CLEAR-P)
4.37	SEARCH	(CLEAR-S)
4.41	VIEW TEXT	(CLEAR-V)

4.46. DELETE BLOCK CLEAR-*

The first time this key is struck the "ONCE MORE TO DELETE" message will flash on and off on the second STATUS LINE. If you change your mind at this point just hit any other key to abort. Hitting CLEAR-* a second time will cause all the text from the cursor to the next BLOCK MARKER character (4.20) to be deleted.

4.47. RETURN CHARACTER ENTER

The RETURN character (←) is entered at the cursor and the cursor is moved to the first column of the next line. Any text, which was to the right of the cursor before the ENTER key was struck, will also move down to the next line.

Use the RETURN character to terminate a paragraph, or a BASIC or an EDTASM statement line, or at any time when you want the text on the line below the cursor to stay separate from the cursored line. If the RETURN character is not used to terminate short lines the LINE JUSTIFICATION function in ZORLOF will, during scrolling, fill each line with as much text that will fit before starting a new line.

Note: The RETURN character is one of the characters which is not displayed while implementing the VIEW TEXT (4.41) or the PRINT TEXT (4.35) functions.

4.48. BLANK CHARACTER CLEAR-SPACE

A BLANK character (▒) is entered at the cursor. This character is special in that it is treated as a hard character (non-space) by ZORLOF's LINE JUSTIFICATION function but as a SPACE by it's PRINT TEXT function. It is quite useful in putting extra spaces between words that the LINE JUSTIFICATION function will not close up and yet is displayed as a SPACE by while VIEWING (4.41) or PRINTING (4.35) text.

Printing in mono-space mode, the width of the BLANK is the same as all the other characters. However, in proportional-space mode, the width of the BLANK will always be 1.25/CHARACTER DENSITY. In other words, if you specify a CHARACTER DENSITY of 10 characters per inch (5.16), a BLANK will be printed .125 inches wide. This width, once established by the CHARACTER DENSITY specification, will never decrease or increase the way a SPACE does in justifying a line to both margins. Therefore, a line that contains many BLANKS and few or no SPACES may actually be too long to justify in proportional-space mode, since each BLANK is 25% larger than the average character is supposed to be for that line. For this reason it is recomended that if printing in proportional-space mode, all long strings of BLANKS in your text alternate one BLANK, one SPACE, one BLANK, one SPACE, etc. If this is done the SPACES will decrease in size to compensate for the oversized nature of the BLANKS, and the line will justify properly.

4.49. END-OF-SENTENCE CHARACTER CLEAR-ENTER

The END-OF-SENTENCE character (▬) is entered at the cursor. Use this character to create an additional amount of space between sentences. This character functions exactly like the BLANK character (▓) except for one respect. If this character falls at the end of a line during PRINTING (4.35) or VIEWING (4.41) and the line justification is JUSTIFY BOTH (2.2, 5.11), this character not only becomes invisible, as a BLANK character would, but the line is rejustified to extend it out to meet the right margin.

4.50. STATUS LINE HOME CLEAR-=

The cursor is moved to the first unprotected column of the top STATUS LINE.

4.51. LEFT BRACKET CHARACTER SHIFT-<

A LEFT BRACKET (<) is entered at the cursor when editing ZORLOF or "ASCII" files. The LESS-THAN (<) sign is entered at the cursor when editing EDTASM or BASIC files.

4.52. RIGHT BRACKET CHARACTER SHIFT->

A RIGHT BRACKET (>) is entered at the cursor when editing ZORLOF or "ASCII" files. The GREATER-THAN sign (>) is entered at the cursor when editing EDTASM or BASIC files.

4.53. RESET CLEAR-0

The first time this key is struck the "ONCE MORE TO CLEAR" message flashes on and off on the second STATUS LINE. IF you change your mind at this point just hit any other key to abort. Hitting CLEAR-0 a second time will cause the screen and the memory to clear and the STATUS LINES to re-initialize.

4.54. SUPERSCRIPT CHARACTER CLEAR-1

An UP ARROW (↑) is entered at the cursor. This character should be used in the text any time a character(s) needs to be printed half a line feed above the base line, like: X^2. The actual character to be SUPERSCRIPTED must be defined in a SUPER/SUBSCRIPT printer command (5.20). For more than one character in a row to be SUPERSCRIPTED, just put that number of UP ARROWS in a row in the text. To SUPERSCRIPT with an expanded character (4.43), just type the expanded character in a SUPER/SUBSCRIPT printer command, then put two up arrows in the text where you want the expanded character to appear.

Note: Of the printers that ZORLOF supports, those that have the capability of printing SUPERSCRIPTS and SUBSCRIPTS are: K2, K3, K5, K6, K7, K8, K9, K10, K11, K12, K13, K14, K15, K16, and K20 (5.1).

4.55. SUBSCRIPT CHARACTER CLEAR-2

This function has the same purpose and operates the same as the SUPERSCRIPT CHARACTER function (4.54) except that the character displayed on the screen is a DOWN ARROW (↓) and is used to locate characters that will be SUBSCRIPTED.

4.56. RENUMBER CLEAR-3

This function only works when the text being edited is a BASIC file or an EDTASM file. The renumbering starts at 10 and counts up by 10's. In both the BASIC and EDTASM files the renumbering is done by stripping off all leading numeric characters on each line, then inserting the appropriate number value. However, EDTASM files will always end up with five-digit line numbers, but the line numbers on BASIC files will only be as many digits as they have to be. For example: Line number 10 will be "00010" in EDTASM files and "10" in BASIC files.

Note: This function will only reassign line numbers. It will not change the GOTO and GOSUB statements in your BASIC text files.

4.57. UNDERLINED SPACE CHARACTER CLEAR-4

An UNDERLINE character (_) is entered at the cursor. Use this character instead of the alternate methods of underlining (4.40, 5.18, 5.27) for when you want just an underline with no character above it as with fill-in-the-bank forms and such.

4.58. ZAP-PROCESSING CLEAR-6

apparently doesn't work in LDOS. make patches in TRSDOS, then copy TO LDOS.

Many of you are computer hackers and will find great value in this unique feature of ZORLOF. Even those of you who are not, will find this feature quite powerful and simple to use, if you choose to take advantage of it. ZAP-PROCESSING as we call it, will allow you to edit (patch) machine code program files by displaying the data in HEX-ASCII format. A portion of what an earlier version of ZORLOF looks like displayed in this format is shown on the next page. The first set 4 digits on each data line is that line's relative address (offset) in the file. The second set of 32 digits is the hexidecimal representation of the 16 bytes at that relative address. The third set of 16 characters is the ASCII representation of those same 16 bytes.

To implement the function just move the cursor to the file name in the directory, like you would to load the file normally (4.25), but this time you hit CLEAR-6. The file will then appear on the screen in "ZAP" format and look something like the example on the next page. You may wish to try this on a program file like ZORLOF/CMD just see it for yourself. Once the file is displayed in this format you may scroll to any place in the file and change anything you want about the file. Only changes made in the data field (middle section), however, will affect the file. You can even get a print-out of the file in this format with CLEAR-P (4.35). When finished, the file can be written back to disk with CLEAR-F (4.24). Deletes have been disallowed in this function to prevent accidental corruption of the file.

From time to time ANITEK may issue patches to ZORLOF; and this is the way you will be able to install them yourself. In Appendix B of this manual there are listed a few patches that deal with items of personal preference which you may make to ZORLOF/CMD if you think they apply to you. Do not make any patches to the master ZORLOF disk, or the master of any other program for that matter. To insure backup, always make patches to a copy of that program.

EXAMPLE OF ZAP FORMAT

```
0000   0506 5A4F 5231 3358 0100 4C59 0000 0000    ..ZDR13X..LY....
0010   0000 0000 0000 0000 0000 0000 0000 0000    ................
0020   0000 20F6 0520 F311 5155 CD13 0020 0FCD    .. v. s.QUM.. .M
0030   3300 FE0D 20F1 CD15 55C2 3040 18E9 FE1C    3.~. qM.U80@.i~.
0040   CA2D 40FE 1DCA 2D40 C34C 554C 494E 4520    J-@~.J-@CLULINE
0050   2054 5900 1151 55CD 1C44 C23A 551A FE2A     TY..QUM.D8:U.~*
0060   CA3A 55CD 3044 CA2D 40C3 4C55 1151 55CD    J:UM0DJ-@CLU.QUM
0070   1C44 C23A 551A FE2A CA3A 5506 0021 0056    .D8:U.~*J:U..!.V
0080   CD24 44C2 4C55 1151 55CD 1300 200C CD3B    M$DBLU.QUM.. .M;
0090   00CD 1555 C230 40C3 C659 F53E 0DCD 3B00    .M.UB0@CFYu>.M;.
00A0   F1FE 1CCA 2D40 FE1D CA2D 40C3 4C55 2100    q~.J-@~.J-@CLU!.
00B0   2100 2100 2100 2100 2100 2100 2100 2100    !.!.!.!.!.!.!.!.
00C0   C33C 5AC3 425A C348 5AC3 4F5A C358 5AC3    C<ZC8ZCHZCDZCXZC
00D0   5E5A C377 5AC3 AD5A C3BF 5AC3 ED5A C31E    ^ZCwZC-ZC?ZCmZC.
00E0   5BC3 295B C32F 5BC3 385B C344 5BC3 4D5B    [C)[C/[C;[CD[CM[
00F0   C357 5BC3 655B C336 5AC3 395A 78B1 C8ED    CW[Ce[C6ZC9Zx1Hm
0100   B0C9 78B1 C8ED B8C9 C54F 0100 4A5A 0600    01x1Hm8IED..JZ..
0110   09C1 C9C5 4F06 00B7 ED42 C1C9 B7ED 4244    .A1ED..7mBA17mBD
0120   4DC9 D5C5 1100 004F 0600 B728 0AB7 ED42    M1UE...D..7(.7m8
0130   3803 1318 F809 7DEB C1D1 C9DD 21AD 5AC5    8...x.}kAQI]!-ZE
0140   DD2B DD2B 0D20 F9C1 3E00 C5DD 4E00 DD23    ]+]+. yA>.E]N.]#
0150   DD46 00DD 233C 12B7 ED42 30F9 3D09 C630    ]F.]#<.7mB0y=.F0
0160   1213 C10D 20E2 C910 27E8 0364 000A 0001    ..A. b1.´h.d....
0170   00D5 C5CD 775A C1E1 0DC8 7EFE 30C0 3620    .UEMwZAa.H~~0@6
0180   2318 F5D5 C506 05E8 2100 001A FE20 2003    #.uUE..k!...~  .
0190   1318 F8CD 1E5B 2016 E60F 0E0A D511 0000    ..xM.[ .f...U...
01A0   EB19 0D20 FC5F 1600 19D1 1305 20DD C1D1    k..  _...Q.. ]AQ
01B0   C97D FE60 D44D 5BD6 30FE 0A38 02D6 07FE    1}~TM[V0~.8.V.~
01C0   1030 1C6F 7CFE 60D4 4D5B D630 FE0A 3802    .0.o ~TM[V0~.8.
01D0   D607 FE10 3009 0707 0707 E6F0 858F C9F6    V.~.0.....fp5?Iv
01E0   FFC9 FE30 D8FE 3A38 02B7 C9BF C9EB CD48    .1~0X~:8.71?1KMH
01F0   5AEB C978 B1C8 7E12 3620 2313 0B18 F4FE    Zk1x1H~.6 #...t~
0200   61D8 FE7B D0D6 20C9 FE41 D8FE 0100 485B    aX~{PV 1~AX~..H[
0210   58D0 C620 C9FE 41D8 FE5B 30E7 C620 C9D5    [PF 1~AX~[0gF 1U
0220   EB21 0000 B72B 0419 3D20 FCD1 C9FE 3038    k!..7(..=  QI~08
0230   14FE 3A38 16FE 4138 0CFE 4738 0EFE 6138    .~:8.~A8.~G8.~a8
0240   04FE 6738 06C5 4704 B8C1 C9BF C905 5CB0    .~g8.EG.8A1?1.\0
0250   3205 5C7E FE29 2867 FE2C 28A6 C339 5CCD    2.\~~)(g~,(&C9\M
0260   D15B C239 5C3A 0D5C F608 320D 5C18 E4CD    Q[89\:.\v.2.\.dM
0270   D15B CA39 5CE5 1155 51CD D150 226A 51E1    Q[J9\e.UQMQP"jQa
0280   0602 18C8 CDD1 5BCA 395C E511 5551 CDD1    ...HMQ[J9\e.UQMQ
0290   5022 6851 E106 0418 B323 7EFE 0DC8 FE29    P"hQa...3#~~.H~)
02A0   C8FE 2CC8 FE3D 20F1 2311 5551 0608 D5C5    H~,H~= q#.UQ..UE
0280   3E20 1213 10FC C1D1 CD81 502B F601 C9ED    > ... AQM.P+v.1m
02C0   4B57 55CD C14A C24C 4061 6263 6465 6667    KWUMAJ8L@abcdefg
02D0   6869 6A68 6C6D 6E6F 7071 7273 7475 7677    hijklmnopqrstuvw
02E0   7879 7A00 0000 0000 3031 3233 3435 3637    xyz.....01234567
02F0   3839 3A3B 2C2D 2E2F 8800 0080 8182 8320    89:;,-./........
0300   0000 0000 0000 0000 6041 4243 4445 0100    ........+MA8CDE.
```

4.59. reserved for future release CLEAR-7

4.60. reserved for future release (Installed 11/9/83). CLEAR-8
Unfreeze-up Key. (Version 2.19H).

4.61. CANCEL KEYBOARD CLICK CLEAR-9

The use of the cassette relay within the expansion interface to produce a gentle click when a key is struck is to give the operator an audible feedback mechanism reassuring him that the key-stroke had been seen and processed. If for any reason you prefer not to take anvantage of this feature you may override it by hitting CLEAR-9.

THIS PAGE INTENTIONALLY

LEFT BLANK

5. PRINTER COMMANDS

PRINTER COMMANDS are used to tell ZORLOF how to format the text that is to printed. Margin size, right and left side text indenting, line spacing, position of header line, and type style are examples of the many PRINTER COMMANDS available to you you.

PRINTER COMMANDS are found in what is known as PRINTER COMMAND LINES. Each PRINTER COMMAND LINE must begin with a PRINTER COMMAND CHARACTER (<)(4.16) and end with a RETURN character (←)(4.47), and each PRINTER COMMAND within the line must be separated from the other PRINTER COMMANDS with commas (,). These PRINTER COMMANDS are visible in the text while the text is being edited, but are stripped out while the text is being PRINTED (4.35) or VIEWED (4.41).

In the following pages, all the various PRINTER COMMANDS are summarized and examples of each are given. An expression showing the standard format of each PRINTER COMMAND is listed to the right of each PRINTER COMMAND name. The upper case characters in the expressions are the "constants", the characters which define the PRINTER COMMAND type, and will not change. The lower case characters in the expressions are the "variables", where you, the user, can specify a value or a string of characters that will be associated with that command from that point on, or until the command is used again somewhere else in the text. Even though, in the standard command format expressions, the constants are upper case and the variables are lower case for clarity in explanation, the constants and the variables of any PRINTER COMMAND can actually be typed in either upper or lower case, whichever is easiest for you.

Even though there are 39 different PRINTER COMMMANDS in all, you need not let that number overwhelm you. ZORLOF is indeed a very simple word processing system to use. You can actually create and print out text without ever using any PRINTER COMMANDS. Try it to prove it to yourself. The internal defaults for each function will take over when the PRINTER COMMANDS are absent. But if there is something special you need a word prcessor to do, it is altogether likely that ZORLOF can do it using one of the following PRINTER COMMANDS.

Caution: There is no error checking on PRINTER COMMANDS. Be careful to follow the examples -- don't leave anything out, don't add anything, and don't change anything, unless it's stated that you can. Improper PRINTER COMMANDS will result in a misinterpretation of that command and often the one following it, as well.

Note: The periods in the examples (...) are to show which commands can be followed by or preceded by other commands, and which ones cannot. They are not part of actual commands.

5.1. PRINTER TYPE Kn

Use this printer command to specify which type of printer you own. This is the only way ZORLOF knows which control codes to use in order to control the various functions of your printer. The printers which are supported by ZORLOF are listed below along with the printer type commands used to specify them. The default for this function is "K0".

K0 Undefined (assumes your printer has no special abilities).

K1 OKIDATA Microline 80, OKIDATA Microline 82A, and OKIDATA Microline 83A (mono-space operation).

K2 Centronics 737, Centronics 739, and Radio Shack Line Printer IV (mono-space operation).

K3 Centronics 737, Centronics 739, and Radio Shack Line Printer IV (proportional-space operation).

K4 EPSON MX-80, MX-80 with Graftrax, EPSON MX-100, and EPSON MX-100 with Graftrax (mono-space operation).

K5 EPSON MX-80 with GraftraxPlus, EPSON MX-100 with GraftraxPlus (mono-space operation).

K6 NEC PC-8023A-C, C. ITOH Prowiter 8510, and Tec 8500R (mono-space operation).

K7 NEC PC-8023A-C (proportional-space operation).

K8 C. ITOH Prowriter 8510 (proportional-space operation).

K9 Tec 8500R (proportional-space operation).

K10 Radio Shack Daisy Wheel II (mono-space operation).

K11 Radio Shack Daisy Wheel II (proportional-space operation).

K12 C. ITOH Starwriter FP-1500, C. ITOH Starwriter F-10, NEC Spinwriter 5515, NEC Spinwriter 5525, Diablo 630, Qume, and most other daisy wheel printers (mono-space operation).

K13 C. ITOH Starwriter FP-1500, NEC Spinwriter 5515, NEC Spinwriter 5525, Diablo 630, Qume, and most other daisy wheel printers (proportional-space operation).

K14 C. ITOH Starwriter F-10 (proportional-space operation).

K15 NEC Spinwriter 5510 and NEC Spinwriter 5520 (mono-space operation.)

K16 NEC Spinwriter 5510 and NEC Spinwriter 5520 (proportional-space operation.)

K17 Brother HR-1 and COMREX ComRiter CR-1 (mono-space operation).

K18 SMITH-CORONA TP-1 (mono-space operation).

K19 Radio Shack Line Printer VI (mono-space operation).

K20 EPSON MX-80 with GraftraxPlus, and EPSON MX-100 with GraftraxPlus (proportional- space operation).

Example: ("..." means possible other commands)

```
<...,K7,...←
```

This command specifies that the text is going to be printed on an NEC PC-8023A-C printer in proportional-space mode.

5.2. AUTOMATIC LINE-FEED INSERTION LF

With ZORLOF you have the option of choosing between having each printed line terminated by a CARRIAGE-RETURN and a LINE-FEED or just a CARRIAGE-RETURN. Which you choose will depend on whether or not your printer can be configured to accept both the CARRIAGE-RETURN and the LINE-FEED for the line termination, and by the printer configuration requirements of your other programs. The default for this function is CARRIAGE-RETURN only.

Note: If your printer is one that is incapable of reverse LINE-FEEDS (K0, K1, K4, and K18; see 5.1) we recommend that this command be used and the printer be configured to accept both the CARRIAGE-RETURN and the LINE-FEED from the host computer. Else, functions that normally require overstriking, such as SUPPRESS LINE-FEED, BOLD, and UNDERLINE, may not work properly on your printer.

Example: ("..." means possible other commands)

```
<...,LF,...←
```

In this example all the text lines will be terminated by sending a CARRIAGE-RETURN and a LINE-FEED to the printer.

5.3. BAUD RATE FOR SERIAL PRINTERS Rn

If this command is specified it is assumed that your printer is connected to the serial port of your computer. If this command is not specified it is assumed that your printer is connected to the parallel printer port on your computer. The 8 different forms of this command and the baud rates which they represent are listed in the table below.

R1	110 baud
R2	150 baud
R3	300 baud
R4	600 baud
R5	1200 baud
R6	2400 baud
R7	4800 baud
R8	9600 baud

Note: The baud rate inside your printer has to be set the same. Consult the operator's manual that came with your printer.

Example: ("..." means possible other commands)

```
<...,R4,...←
```

This command specifies a serial printer running at 600 baud, rather than a parallel printer.

For smoothest and fastest printing, we recommend that you set your printer at the highest BAUD rate that it is capable of running at.

5.4. SETTING OF TABS TABa;b;c;d;e;f.....

This printer command is used to set the tab stops for both editing and printing. The letters in the expression (a-f...) are the tab stop locations relative to the left most column of the line. These must be specified in ascending order within the command. As many as 15 tab stops can be specified. This command will become active on the screen for editing once entered and CLEAR-T (4.39) has been struck. The default for this command is:

```
<TA86;14;22;30;38;46;54;62;70;78;86;94;102;110;118←
```

Note: All the stops within the command are separated by semicolons ";" **not** commas.

Caution: For proper operation, no more than one TAB STOP command sould be made per file and it should be located within the text file ahead of any printed text.

Example: ("..." means possible other commands)

```
        <...,TAB10;17;21;28;35;42;49;56,...←
```

The first tab stop is 10 columns in from the left side, the second is 17 columns in from the left side, ..., the last one is 56 columns in from the left side.

5.5. INDIVIDUAL SHEET PAUSING

If you are feeding your printer with individual sheets instead of a continuous roll or a box of fan fold paper, then you will want to use this command. Each time the print head reaches the line number on the page that you have specified as the length of the paper (5.6), the printing will pause. This gives you the opportunity to remove that sheet from the printer and put a new one in without losing any text. To restart the printer at the point where you left off, either hit the space bar on the keyboard or turn the on-line switch off and back on. This action will have to be taken before ZORLOF will continue printing any sheet of that file - <u>even the first sheet</u>. The default for this function is that the individual sheet pausing is not active.

Caution: If you own a serial printer and want to feed it individual sheets it is necessary that the command specifying the serial printer (5.3) comes before this command or improper operation will result.

Example: ("..." means possible other commands)

```
<...,I,...←
```

This command activates the single sheet pausing function.

5.6. SHEET SIZE SET Yn

This command is used to set the SHEET SIZE of the paper on your printer. The "n" in the expression is the length in number of lines. The default for this function is "Y66".

Example: ("..." means possible other commands)

```
<...,Y48,...←
```

This specifies a SHEET SIZE of 48 lines long, or 8 inches long assuming 6 lines per inch spacing (standard).

5.7. TOP TEXT LINE SET Tn

This printer command establishes on which line, counting from the top of the page, the body of the text will begin. The "n" in the expression is that line number, 1 being the very top of the page. The default for this function is "T7".

Caution: The number that you assign for the TOP TEXT LINE must be less than or equal to the number you assign for the END TEXT LINE (5.8), and must be greater than 0.

Example: ("..." means possible other commands)

```
<...,T9,...←
```

In this example the text body will begin approximately 1.5 inches below the top of the page, assuming 6 lines per inch spacing (standard).

5.8. END TEXT LINE SET En

This command is just like the TOP TEXT LINE SET printer command (5.7), except that it specifies on which line the last line of the text body will be printed. The default for this function is "E62".

Caution: The number that you assign for the END TEXT LINE must be greater than or equal to the number you assign for the TOP TEXT LINE (5.7), and must be less than or equal to the number you assign for the SHEET SIZE (5.6).

Example: ("..." means possible other commands)

```
    <...,E57,...←
```

In this example the text body will end approximately 9.5 inches below the top of the page, assuming 6 lines per inch spacing (standard).

Example: ("..." means possible other commands)

```
    <...,T1,E1,Y1,...←
```

This set of commands will allow you to print out any number of text lines with out any paper waste at the top or bottom (no top or bottom margin.

5.9. TEXT LEFT MARGIN MOn,MEn

This printer command will consist of either "MO" followed by a value "n" to specify the width of the left side margin on odd numbered pages, or "ME" followed by a value "n" to specify the width of the left margin on the even numbered pages. Both must be specified or the default (MO8,ME8) will be assumed. The unit of measure for "n" is either tenths of an inch if you are printing in proportional-space mode, or the unit of measure is character widths if you are printing in mono-space mode.

Example: ("..." means possible other commands)

```
<...,MO12,...,ME6,...←
```

In this example, the left margin width is 1.2 inches on the odd numbered pages and .6 inches on the even numbered pages, if you are printing in proportional-space mode. Otherwise, the left margins in this example will be 12 character widths and 6 character widths, respectively. The actual width of the character depends on the type of printer you own and whether you are printing in EXPANDED (5.13), CONDENSED (5.14), or NORMAL (5.15) width character mode. Consult the operator's manual that came with your printer for this information.

5.10. LINE SPACING   Ln

The "n" in this expression is to be set to the desired spacing of the text lines -- "1" for single spaced-lines, "2" for double-spaced, etc. No value for "n" less than 1, greater than 10, or greater than the number you specified as the SHEET SIZE (5.6) is allowed. The default for this function is "L1".

Note: Improper matching of the AUTOMATIC LINE-FEED INSERTION printer command (5.2) with the configuration of your printer will result in improper line spacing. If you are getting double spacing when you specified single spacing, it is probably because you are using the "LF" command (5.2) and shouldn't be.

Example: ("..." means possible other commands)

```
<...,L2,...←
```

In this example the text lines will be printed double-spaced.

5.11. JUSTIFY MODE SET Jx

This printer command lets you specify which of 4 possible LINE JUSTIFICATION modes you want the text to be printed in.

JL Justify text lines to the left margin.

JR Justify text lines to the right margin.

JB Justify text lines to both the left and right margins.

JC Center the text between the left and right margins.

These 4 different commands will not only dictate the way the text will justify on the page as it is being PRINTED (4.35), but will also cause the text to justify accordingly on the screen while the text is being VIEWED (4.41), as well as while you are editing it. The line containing the command must, however, be scrolled off the screen then brought back onto the screen by scrolling up in order to set the LINE JUSTIFICATION mode for that portion of text while you are editing it. Hitting CLEAR-T (4.39) is one way of achieving these results. The default JUSTIFY mode for PRINTING, VIEWING, and editing is "JL".

Note: While PRINTING or VIEWING text that is set to JUSTIFY RIGHT (JR), lines terminated by RETURN characters (←) will be justified fully to the right margin after the RETURN character has been stripped.

Example: ("..." means possible other commands)

```
<...,JL,...←
Mares eat oats,←
<...,JR,...←
And does eat oats,←
<...,JB,...←
And little lambs
<...,JC,...←
eat ivy.←
```

This will be printed:

```
Mares eat oats,
    And does eat oats,
And    little    lambs
       eat ivy.
```

5.12. LEFT AND RIGHT SIDE TEXT INDENTING ILn,IRn

These two printer commands can be used to achieve text widths which are less than the maximum (set in the WIDTH FIELD, see 3.2), by specifying an additional amount of indenting on the left or right sides or both. The "n" in the expressions is the number of character widths of indenting desired. This is often useful in highlighting certain paragraphs by causing them to be narrower than the rest of the text body. Full justification is maintained on all indented portions of text. The default for this function is "IL0,IR0".

Note: The LEFT SIDE TEXT INDENT is in <u>addition</u> to the TEXT LEFT MARGIN as described in section 5.9.

Example: ("..." means possible other commands)

```
▓▓▓The following letter was received  by  our
branch office in Chicago.←
<...,IL5,IR5,GR1,...←
     Dear Sirs,←
     ▓▓▓Thank   you  for  your  help  in
     locating the parts that I needed. I
     will  be  doing  more business with
     you in the future.←
                   Sincerely,←
                   Jack Thomson←
<...,IL0,IR0,GR1,...←
▓▓▓Congratulations  Chicago!   It's  happy
customers  like  this  one  that have kept us
going strong.←
```

This will be printed:

```
    The following letter was received by our
branch office in Chicago.

     Dear Sirs,
         Thank you for your help in
     locating the parts that I needed. I
     will be doing more business with
     you in the future.
                         Sincerely,
                         Jack Thomson

    Congratulations Chicago!  It's happy
customers like this one that have kept us
going strong.
```

5.13. EXPANDED WIDTH CHARACTERS W

Use this printer command if you want your text to be printed in EXPANDED width character mode, provided your printer has this capability. If your printer is type K2, K3, K4, K5, K6, K7, K8, K9, K19, or K20 (5.1), you can also print in what is known as CONDENSED-EXPANDED character mode. To achieve this type style just use this printer command along with the CONDENSED width character printer command (5.14). Terminate both the EXPANDED and the CONDENSED width character modes with the NORMAL width character mode printer command (5.15). The default for this function is NORMAL width characters.

Note: If you wish to mix EXPANDED width characters and NORMAL width characters on the same line is it would be better to use the EXPANDED CHARACTER function discribed in section 4.43.

Example: ("..." means possible other commands)

```
<...,W,...←
John Doe←
<...,N,...←
is my name.←
```

This will be printed:

```
J o h n   D o e
is my name.
```

5.14. CONDENSED WIDTH CHARACTERS Q

Use this printer command if you want your text to be printed in CONDENSED WIDTH CHARACTER mode, provided your printer has this capability. If your printer is type K2, K3, K4, K5, K6, K7, K8, K9, K19, or K20 (5.1), you can also print in what is known as CONDENSED-EXPANDED character mode. To achieve this type style just use this printer command along with the EXPANDED width character printer command (5.13). Terminate both the CONDENSED and the EXPANDED width character modes with the NORMAL width character mode printer command (5.15). The default for this function is NORMAL width characters.

Example: ("..." means possible other commands)

```
<...,Q,...←
My name is←
<...,N,...←
John Doe.←
```

This will be printed:

```
My name is
John Doe.
```

5.15. NORMAL WIDTH CHARACTERS

N

This printer command is used to cancel the EXPANDED and/or CONDENSED width character modes (5.13, 5.14). The default for this function is NORMAL width character mode.

Example: (see 5.13 or 5.14)

5.16. CHARACTER DENSITY Dn

This printer command is only needed if you are printing in proportional-space mode. This command gives you the capability of deciding how tightly packed you want a line of characters to be printed. The "n" in the expression is the CHARACTER DENSITY, in characters per inch, and can be any value from 6.0 to 20.0, in increments of .1 characters per inch.

From this density figure you can determine the actual length of the line in inches knowing the length of the line in characters. Say you have specified your text lines to be 64 characters long, max (3.2), and you want that to print out to exactly 5 inches. Using a little simple math you would determine that this translates to a CHARACTER DENSITY of 12.8 characters per inch. The default for this function is 13.0 characters per inch for printer types K3, K7, K8, and K9; and 10.0 characters per inch for printer types K11, K13, K14, K16, K20 (5.1). These are, by the way, just about as tightly as you normally can pack a line on each of the given printers and still have some noticeable amount of space between the words. This manual was printed with a CHARACTER DENSITY of 11.0 characters per inch.

Note: As long as you are printing in NORMAL or CONDENSED WIDTH CHARACTER mode, the density you specify will be the actual density printed. EXPANDED WIDTH characters will be printed at half the density you specify.

Caution: If character densities are specified which exceed 10.5 on daisy wheel printers or 14.0 on dot matrix printers, you may see one or more lines in which the characters did not justify properly. If this should ever happen it will be necessary to specify a lower value for the character density.

Example: ("..." means possible other commands)

```
<...,D13.8,...←
Jack and Jill ran up a hill.←
<...,D6.5,...←
Jack and Jill ran up a hill.←
```

This will be printed:

```
Jack and Jill ran up a hill.
J a c k   a n d   J i l l   r a n   u p   a   h i l l .
```

5.17. BOLDING TEXT Bxxxx...

Using this command, you can specify which characters in the following text line you want to be bolded. If that string of characters is found in the following text line, those characters will be over-printed 7 times to cause them to appear darker than the other characters which are only printed once. The "xxxx..." in this expression is that string of characters and can be as long or as short as you wish, provided all the characters of the string fit on the same printer command line. Up to 2 BOLD printer command strings can be specified per text line.

If your printer is a type K4, K5, or K20 (5.1), you are better off in this regard. These printers have a DOUBLE PRINT--EMPHASIZE mode which can be used to achieve bold printing. If you have specified the K4, K5, or K20 printer type in a previous printer command, ZORLOF will automatically take advantage of these printers' special abilities so your BOLD can be achieved by only 1 pass over the text line instead of 8, as required on the other types of printers.

Note: If the BOLDED text is also specified to be UNDERLINED (4.40, 5.18), the UNDERLINE will also be BOLDED.

Note: For proper registration of the overprinted text during BOLDING, it is recomanded that the fiction feed mechanism be engaged at all times, even on tractor feed printers.

Caution: No other printer command should follow the BOLD printer command on the same line, since commas (,) in this printer command are considered to be part of the text that will be BOLDED, and not as the delimiters between commands as they usually are.

Example: ("..." means possible other commands)

```
<...,Bnot←
<...,Bhumid←
The unit will not operate in humid conditions.←
```

This will be printed:

The unit will **not** operate in **humid** conditions.

5.18. UNDERLINING TEXT Uxxxx...

Using this command, you can specify which characters in the following text line you want to be UNDERLINED. If that string of characters is found in the following text line those characters will be printed with an UNDERLINE. The "xxxx..." in this expression is that string of characters and can be as long or as short as you wish, provided all the characters of the string fit on the same printer command line. Up to 2 UNDERLINE printer command strings can be specified per text line.

If your printer is a type K2, K3, K5, K6, K7, K8, K9, K10, K11, K14, K18, or K20 (5.1), you are better off in this regard. These printers have the capability of printing the underline at the same time as the text. As long as you made the right PRINTER TYPE specification (5.1), ZORLOF will take advantage of this feature and give you UNDERLINING with only 1 pass over the text line instead of 2, as required on the other types of printers.

Note: If the UNDERLINED text is also specified to be BOLDED (5.17), the UNDERLINE will also be BOLDED.

Caution: No other printer command should follow the UNDERLINE printer command on the same line, since commas (,) in this printer command are considered to be part of the text that will be underlined, and not as the delimiters between commands as they usually are.

Example: ("..." means possible other commands)

```
<...,Unot←
<...,Uhumid←
The unit will not operate in humid conditions.←
```

This will be printed:

```
The unit will not operate in humid conditions.
```

(See sections 4.40, 4.57, and 5.27 for alternate methods of achieving underlined text.)

5.19. ITALIC TEXT Vxxxx...

Using this command, you can specify which characters in the following text line you want to be printed in ITALICS. The "xxxx..." in this expression is that string of characters and can be as long or as short as you wish, provided all the characters of the string fit on the same printer command line. Up to 2 ITALIC printer command strings can be specified per text line.

Of the printers that ZORLOF supports, the Epson MX-80 and MX-100 with Graftrax or Graftrax Plus (K5 or K20, see 5.1) are the only printers that have the capability of directly printing text in ITALICS. ITALIC printing can also be accomplished on any printer that has the capability of high resolution dot graphics (K2, K3, K6, K7, K8, and K9), through the use of the SPECIAL CHARACTER function described in section 5.27. Although it is a bit more work, the results can be quite impressive. The example below was printed using this method.

Caution: No other printer command should follow the ITALIC printer command on the same line, since commas (,) in this printer command are considered to be part of the text that will be printed in ITALICS, and not as the delimiters between commands as they usually are.

Example: ("..." means possible other commands)

```
<...,Vnot←
<...,Vhumid←
The unit will not operate in humid conditions.←
```

This will be printed:

The unit will *not* operate in *humid* conditions.

This command is used in conjunction with the SUPERSCRIPT character (↑)(4.54) and SUBSCRIPT character (↓)(4.55) to achieve SUPERSCRIPTS and SUBSCRIPTS in your printed text. In this expression the "xxxx..." is a string of characters that defines which characters are to be SUPERSCRIPTED and SUBSCRIPTED. You can use one printer command to define all the SUPER/SUBSCRIPTS in the entire text if you wish, or you can use several shorter ones as they are needed throughout the text. In either case, there must be one character defined by this command for every SUPERSCRIPT character (↑) or SUBSCRIPT character (↓) that appears in the text -- no more and no less.

Note: It is not possible to have SUPER/SUBSCRIPTED text BOLDED, or in ITALICS, unless done through the use of the SPECIAL FUNCTION printer command (5.27).

Note: For proper registration of the SUPERSCRIPTED and SUBSCRIPTED characters, it is recomended that the friction feed mechanism be engaged at all times, even on a tractor feed printer.

Caution: This function will only work on K2, K3, K5, K6, K7, K8, K9, K10, K11, K12, K13, K14, K15, K16, and K20 printer types (5.1).

Example: ("..." means possible other commands)

```
<...,S123(x+1),...←
X↓+X↓+X↓=Y↑↑↑↑↑←
```

This will be printed:

$X_1+X_2+X_3=Y^{(x+1)}$

5.21. SUPPRESS LINE-FEED

Use this printer command if you ever want to over-print one line with another line. This command only affects the text line that immediately follows it and none other.

Example: ("..." means possible other commands)

```
<...,XL,....←
The current is not to exceed 6.300 amps.←
▓ ▓ ▓ ▓ ▓ ▓ ▓ ▓ ▓ ▓ ▓ ▓ ▓ ▓ ▓ ▓ //←
```

This will be printed:

```
The current is not to exceed 6.3ØØ amps.
```

5.22. SUPPRESS CARRIAGE-RETURN XC

Use this printer command to join 2 or more display lines to achieve an extra-long printed line. This command only affects the text line that immediately follows it and none other.

Note: The LINE-FEED for the first line and the TEXT LEFT MARGIN for the second line are also suppressed automatically at the time this command is executed.

Caution: No portion of the second (or higher) adjoining line segments can contain any BOLDED, or ITALIC characters, unless done through the use of the SPECIAL FUNCTION printer command (5.27).

Example: ("..." means possible other commands)

```
<...,XC,...←
Service is our middle name, quali
ty our first.←
```

This will be printed:

```
Service is our middle name, quality our first.
```

5.23. TURN PRINTING OFF AND ON OFF,ON

Use these two printer commands in cases when you have a very large text file, but you want only certain parts of it to be printed. These commands can be used to delimit comment statements throughout your text, as well.

All printer commands that fall between the "OFF" and "ON" commands, except for PRINTER CONTROL CODES (5.26), will be processed, even though the text is treated as if it wasn't there.

Note: Using the "ON" and "OFF" printer commands will not cause gaps in text while the text is being PRINTED on the page or VIEWED on the screen. The delimited text is treated like it wasn't even there.

Example: ("..." means possible other commands)

```
<...,OFF,...←
Since the meeting on Tuesday, orders have
been on a steady increase.←
<...,ON,...←
```

In this example the sentence between the "OFF" and the "ON" commands is treated, as far as the PRINT TEXT and the VIEW TEXT functions are concerned, as if it was not in the text.

5.24. GO TO LINE NUMBER Gn

Upon receipt of this command ZORLOF will hold off sending any text to the printer. Instead, line feeds will be sent to the printer until the print head reaches the line number on the page specified by the "n" in this command. At which time the printing of the text resumes. This command is very useful for leaving a large blank area on the paper without putting a lot of RETURN characters in your text. If the print head passes a HEADER (5.28), FOOTER (5.29), or PAGE NUMBER LINE (5.30) on the way to its "Go To" destination, it (they) will be printed.

Example: ("..." means possible other commands)

```
<...,G1,...←
```

In this example the text will resume printing at the top of the text body on the next page. HEADER, FOOTER, and PAGE NUMBER LINES will be printed.

5.25. GO RELATIVE GRn

This printer command is very similar to the GO LINE NUMBER command (5.24) except that it allows you to specify a desired number of blank lines without knowing what line number you are currently on.

Example: ("..." means possible other commands)

```
      Peace on Earth,←
      <...,GR3,...←
      And good will toward men.←
```

This will be printed:

```
      Peace on Earth,



      And good will toward men.
```

Most printers can do more than just print characters. Some can roll the paper backwards, set vertical tabs, change type styles, etc. These are all controlled and communicated to the printer by what is known as PRINTER CONTROL CODES (ASCII values less than 20H). If you look in the operator's manual that came with your printer you will find a list of these special capabilities and the PRINTER CONTROL CODES used to make them work.

If you own one of the printers listed in section 5.1 and have used the proper PRINTER TYPE specification, you can have access to most of these special capabilities through the use of the appropriate printer commands. However, if you have a different model printer, or just want to have direct control of your printer's special abilities, ZORLOF gives you the freedom to do so using this printer command. All of the special type fonts of the Epson printers can be achieved using this function (see second example).

The "aabb..." in this expression is a set of one or more (not more than a line full) two-digit hexidecimal numbers (with no separation). The "Xn" combination in this expression is optional and only needed if you wish the string of PRINTER CONTROL CODES to be sent to the printer a repeated number of times. The "n" is any decimal value from 2 to 65535. The "L" in the second expression indicates that this string of CONTROL CODES should be sent to the printer "n" number of times at the start of <u>every</u> line. Each "CL..." printer command in the text will cancel any preceding "CL..." commands.

Example: ("..." means possible other commands)

```
<...,C0AX20,...←
```

This command will send 20 line feeds (0AH) to the printer.

```
<...,CL1B45,...←
```

On an Epson printer, this command will cause all of the following text to be printed in the "Emphasized" type font.

5.27. SPECIAL FUNCTION CHARACTER £(n)'xaa'y'zbb...

The SPECIAL FUNCTION CHARACTER printer command is in many ways similar to the Printer CONTROL CODE printer command (5.26). However, it goes one step further, in that it allows you to send to the printer a string of ASCII characters and printer control codes mid-line. One SPECIAL FUNCTION character is used in the line of text to "locate" the string (4.17), while a previous SPECIAL FUNCTION printer command "defines" that character.

The "n" in the expression is an optional width value for the character thus defined, and is only necessary if printing in proportional-space mode with text justified to the right and left margins. This tells ZORLOF how much room, in proportional-space pixel widths, this special character is going to require. Any value for "n" from 0 to 254 is allowed. If the "(n)" is used, it must follow right after the SPECIAL FUNCTION character (£) within the printer command line. If the "(n)" is omitted from the expression, then the normal proportional-space width of the first ASCII character found in the string is assumed. If there are no ASCII characters in the string, a width of 0 is assumed. The widths of measure for "n" (dot widths) on the various printer types are:

1/160	inches	K7, K8, K9
1/150	inches	K3
1/120	inches	K13, K14, K16, K20
1/60	inches	K11

The "'x", "'y", and "'z" in the expression are the ASCII characters. Any single character preceded by an apostrophe (') is considered in this command to be an ASCII character. The "aa" and "bb" in the expression are two-digit hexidecimal numbers. These will most often be printer control codes which cannot be expressed as ASCII characters. However, even ASCII characters can be expressed as two-digit hexidecimal numbers if you choose. These can be located in any order and in any amount throughout the expression.

This is the only printer command that can be more than one display line long. See the last example in this section. Follow these rules for SPECIAL FUNCTION printer commands that have to be longer than one display line:

1. Each SPECIAL FUNCTION printer command line should start with the PRINTER COMMAND character (<), but only the first line should contain the SPECIAL FUNCTION character (£).
2. Only the last line of the printer command should be terminated by a RETURN character (←) or a comma (,).
3. Do not break the line in a place that would split a pair of characters which represents an ASCII character ('x) or a printer control code (aa).

The following are some examples of how the SPECIAL FUNCTION character can be used. Keep in mind that each one of these examples was printed using a NEC PC-8023A-C printer. If you have a different type of printer, the concept will be the same, but the implementation will have to conform to your own printer's specifications. Study your printer manual to learn the printer control code representations of graphic characters, extended symbols, foreign language characters, control of type style enhancements, fonts, and the control of print head and carriage movements. Once learned, you could be using the SPECIAL FUNCTION command to create countless symbols, shapes, borders, pictures -- you name it.

1. Access to foreign characters:

```
<£1B26'J1B24←
The circumference of a circle is 2£.←
```

printed:

```
The circumference of a circle is 2π.
```

2. Access to symbols:

```
<£1B23'11B24,£1B23'H1B24←
A bid of 3£ cannot follow a bid of 3£.←
```

printed:

```
A bid of 3♥ cannot follow a bid of 3♠.
```

3. Access to print enhancements.

```
<£1B5B'u,£'e1B59←
This is another way to £nderlin£ a word.←
```

printed:

This is another way to <u>underline</u> a word.

4. Access to dot graphics (justified in proportional-space).

```
<£(25)1B'S'0'0'2'54B002400120009001200240 04B0024
<001200090012002400 4B←
<£(27)1B'S'0'0'2'71C00140014001400140014001400 14
<0022004100410041002200 1C←
     While on vacation in Africa I discovered a
small piece of pottery with two strange marks on
it.  One was a wavy set  of  lines  £,  and  the
other  was  a circle with a kind of handle £.  I
have been looking for for a year for someone who
can tell me its meaning.←
```

printed:

While on vacation in Africa I discovered a small piece of pottery with two strange marks on it. One was a wavy set of lines ≋, and the other was a circle with a kind of handle ⟜. I have been looking for for a year for someone who can tell me its meaning.

5.28. HEADER LINE SET Hn

This command sets the vertical positioning of the user formattable HEADER LINES. This is usually specified to fall somewhere between the top of the page and the top of the text body, but could also be specified to fall below the text body, if desired. However, it should never be specified to fall within the text body (5.7, 5.8), or on the same line as the FOOTER (5.29) or PAGE NUMBER LINE (5.30), or it may not get printed. The function can be deactivated altogether by using the command: "H0". The default for this command is "H0".

Example: ("..." means possible other commands)

```
        <...,H3,...←
```

In this example the HEADER LINES will be printed on the third line of the paper, or about a half an inch below the top of the page, assuming 6 lines per inch spacing (standard).

5.29. FOOTER LINE SET Fn

This printer command is just like the HEADER LINE SET command (5.28) except that it sets the vertical positioning of the user formattable FOOTER LINES. This is usually specified to fall somewhere between the end of the text body and the bottom of the page, but could also be specified to fall above the text body to achieve a double HEADER LINE, if desired. However, it should never be specified to fall within the text body (5.7, 5.8), or on the same line as the HEADER (5.28) or PAGE NUMBER LINE (5.30), or it may not get printed. The function can be deactivated altogether by using the command: "F0". The default for this command is "F0".

Example: ("..." means possible other commands)

```
<...,F63,...←
```

In this example the FOOTER LINES will be printed on the 63rd line of the paper, or about a half an inch from the bottom of the page, assuming 11 inch long paper and 6 lines per inch spacing (standard).

5.30. PAGE NUMBER LINE SET Pn

This printer command is just like the HEADER LINE SET command (5.28) except that it sets the vertical positioning of the user formattable PAGE NUMBER LINES. This is usually specified to fall somewhere between the end of the text body and the bottom of the page, but could also be specified to fall above the text body, if desired. However, it should never be specified to fall within the text body (5.7, 5.8), or on the same line as the HEADER (5.28) or FOOTER LINES (5.29), or it may not get printed. The function can be deactivated altogether by using the command: "P0". The default for this command is "P0".

Example: ("..." means possible other commands)

```
<...,P4,...←
```

In this example the PAGE NUMBER LINES will be printed on the 4th line of the paper, or about a half an inch from the top of the page, assuming 6 lines per inch spacing (standard).

ZORLOF allows you to define 2 different HEADERS -- one for the odd-numbered pages and one for the even-numbered pages. The full command consists of either "HO" or "HE" to specify which HEADER is being defined (odd or even), followed by "JL", "JR", "JB", or "JC" to specify how the line will be justified (5.11), and followed finally by a RETURN character (←). The "Jx" part of the expression is optional; if not used, "JR" is the default for odd-numbered pages, and "JL" for even-numbered pages.

On the line immediately following this command you must type the HEADER LINE just as you would like to see it printed on the paper. Define the HEADER LINE for the alternately numbered pages in the same way. It is necessary to define both headers if they are going to be used at all, even if they are exactly the same. If you do not wish to implement HEADER LINES you can deactivate this function by setting HEADER LINE SET command to "H0" (5.28).

Note: It is possible to have UNDERLINED (4.40) and EXPANDED (4.43) characters in the HEADER LINE, but not BOLDED, CONDENSED, or ITALIC characters.

Caution: The HEADER LINE DEFINE FORMAT line must be followed by a RETURN character (←), even if it must be placed on the following line. If the RETURN character is accidentally omitted, the text from the HEADER LINE definition to the next occurrence of a RETURN character in your text will not be printed.

Example: ("..." means possible other commands)

```
<...,HOJC←
¦T¦o¦p¦i¦c¦a¦l▓▓¦R¦e¦v¦i¦e¦w←
<...,HEJC←
A Space In Time←
```

In this example the HEADER LINE on the odd-numbered pages will be printed:

Topical Review

For the even numbered pages:

A Space In Time

5.32. FOOTER LINE FORMAT FOJx,FEJx

The explanation for this command is the same as for the HEADER LINE FORMAT printer command (5.31).

5.33. PAGE NUMBER LINE FORMAT POJx,PEJx

The explanation for this command is the same as for the **HEADER LINE FORMAT** printer command (5.31), except that you can put, somewhere in the line definition, a string of one or more **NUMBER SYMBOL** characters (#) which will be replaced by the current line number when the line is printed. Use enough **NUMBER SYMBOL** characters for the amount of digits of your highest page number.

Example: ("..." means possible other commands)

```
<...,POJB←
Roses In Springtime ▒ ▒ ▒ ▒ ▒ ▒ ▒ ▒ ▒ Page ###
←
<...,PEJ8←
Page ### ▒ ▒ ▒ ▒ ▒ ▒ ▒ ▒ ▒ Roses In Springtime
←
```

The **PAGE NUMBER LINE** for page 1 will be printed:

```
Roses In Springtime                  Page   1
```

The **PAGE NUMBER LINE** for page 248 will be printed:

```
Page 248                  Roses In Springtime
```

5.34. HEADER LINE LEFT MARGIN HMOn,HMEn

This command is just like the TEXT LEFT MARGIN printer command (5.9), and all the same rules, explanations, defaults, and examples apply. The only difference is that this command specifies the left side margin for the HEADER LINES instead of the TEXT LINES.

5.35. FOOTER LINE LEFT MARGIN FMOn,FMEn

This command is just like the TEXT LEFT MARGIN printer command (5.9), and all the same rules, explanations, defaults, and examples apply. The only difference is that this command specifies the left side margin for the FOOTER LINES instead of the TEXT LINES.

5.36. PAGE NUMBER LINE LEFT MARGIN PMOn,PMEn

This command is just like the TEXT LEFT MARGIN printer command (5.9), and all the same rules, explanations, defaults, and examples apply. The only difference is that this command specifies the left side margin for the PAGE NUMBER LINES instead of the TEXT LINES.

5.37. PAGE NUMBER SET P#n

This printer command sets the starting page number. It can be changed at any place in the text if desired. The default for the function is "P#1".

Example: ("..." means possible other commands)

```
<...,P#100,...←
```

All pages printed after this command will be numbered counting from 100.

Caution: It is usually best to position the PAGE NUMBER SET command at a point in the text where it will be processed between the printing of the last line of one page and the first line of the next page. The only way to insure this is to put the PAGE NUMBER SET command in a printer command line ahead of the first text line to set the page number initially, and on the very next line following a GO TO LINE NUMBER command (Gn, see 5.24) where "n" in the expression is the number of lines per page that you set by the PAGE LENGTH SET command (5.6). Do not put both on the same printer command line or it will not work correctly.

Example: ("..." means possible other commands)

```
<...,G66,...←
<...,P#281,...←
```

The "G66" command will cause the current page to finish printing, assumimg the page length was set at 66 lines, and the following page will start numbering at 281.

5.38. FORM LETTER DATA FILE SPECIFICATION DATAFILE=filename

Use this printer command at the very top of a FORM LETTER TEXT FILE to tell ZORLOF the file name of the FORM LETTER DATA FILE containing the data records that will be used as input to the FORM LETTER TEXT FILE at the time of the form letter printing run. See section 7 for a complete explanation of how to create and run FORM LETTER TEXT FILES and FORM LETTER DATA FILES.

The "DATAFILE=" in this expression is the constant, but can, however, be shortened to "DA=" if you wish. The "filename" in this expression is the real name that you have given to the FORM LETTER DATA FILE to be used with this FORM LETTER TEXT FILE.

If this printer command is present, then, at the time of the PRINTING (4.35) or VIEWING (4.41) of the FORM LETTER TEXT FILE, ZORLOF will load the FORM LETTER DATA FILE into memory one data record at a time, replace the data labels within the FORM LETTER TEXT FILE with the data fields associated with those labels from each data record of the FORM LETTER DATA FILE. Once all the replacing is finished for the first data record, the text is printed, and the process is repeated, giving one printed letter (document, lease, whatever) for each data record of the FORM LETTER DATA FILE. If this printer command is <u>not</u> present, ZORLOF assumes that the text file is not a FORM LETTER TEXT FILE, and at the time of PRINTING or VIEWING will print out the text file only once and without any replacement of the data labels.

Example: ("..." means possible other commands)

```
<...,DATAFILE=FORM/DAT,...←
```

This printer command tells ZORLOF that the text file in memory is a FORM LETTER TEXT FILE, containing data labels that will be replaced by data fields, which are contained in data records, which are found on the disk under the file name of "FORM/DAT".

5.39. FORM LETTER DATA ID QUALIFIER PRINT=abcde...

This is an optional printer command used within a FORM LETTER TEXT FILE to specify which data records from the FORM LETTER DATA FILE will be used as data inputs during the form letter printing run. Only those data records that have ID's matching one of those specified in this printer command, and data records which have no ID's at all, will be printed. If this command is omitted from the FORM LETTER TEXT FILE, there will be one copy of the FORM LETTER TEXT FILE printed for every data record of the FORM LETTER DATA FILE, regardless of the records' ID character.

The "PRINT=" is the constant in this expression, but can be shortened to "PR=" if you wish. The "abcde..." is a list of one character data record ID's (no separation) Which are to match the one character ID's of the data records contained within the FORM LETTER DATA FILE, that you want to be used as input to the FORM LETTER TEXT FILE at the time of the run. The data record ID can be any character except the SPACE ( ) and the RETURN character (←). If used, this command must be positioned in the FORM LETTER DATA FILE ahead of any printed text. See section 7 for a complete explanation of how to create and run FORM LETTER TEXT FILES and FORM LETTER DATA FILES.

Example: ("..." means possible other commands)

```
<...,DATAFILE=FORM/DAT,PRINT=XQT,...←
```

This printer command tells ZORLOF that the text file in memory is a FORM LETTER TEXT FILE, containing data labels that will be replaced by data fields, which are contained in data records, which are found on the disk under the file name of "FORM/DAT". <u>But</u>, only those records that have an ID of X, Q, T, or no ID at all will be used. The other data records contained within FORM/DAT will be skipped.

6. EDITING BASIC AND EDTASM FILES

Almost every capability available to you while editing normal text files can also be used when editing BASIC and EDTASM files. A few things work a little differently due to the unique nature of these types of files. And then there are an additional couple of capabilities ZORLOF reserves just for these two file types. We are sure that as you become familiar with these few special provisions you will soon be using the same high-speed, labor-saving powers of ZORLOF to edit your program source files as you have for your text files.

6.1. WIDTH FIELD

The contents of the WIDTH FIELD (3.2 and Appendix A) will determine whether ZORLOF treats your text as normal text or as a BASIC or EDTASM file. If the WIDTH FIELD contains "BAS", the text is treated as a BASIC file. If the WIDTH FIELD contains "EDA", the text is treated as an Apparat EDTASM file. If the WIDTH FIELD contains "EDR", the text is treated as a Radio Shack EDTASM file.

6.2. INSERT PROGRAM LINE

When the OPEN LINE function (CLEAR-O, see 4.34) is implemented while editing a BASIC or EDTASM file, a special thing happens: ZORLOF goes into the INSERT PROGRAM LINE mode. All the text below the cursor moves down one line, a new line number is automatically written at the start of the new blank line (one greater than the value of the previous line number), and the cursor is moved to the second space after the line number. While in this mode, each time a line is ended by a RETURN character (←)(striking the ENTER key, 4.47) another blank line with a new line number appears on the screen.

This function will terminate automatically when the line number of the current line is one less than the line number of the following line, or manually by moving the cursor to the start of the line and hitting CLEAR-L (4.31).

6.3. RENUMBERING

Upon hitting CLEAR-3 the BASIC or EDTASM file is removed from the screen, all the program lines are renumbered starting at 10 and counting by 10, and the text returns to the screen. This function is also covered in section 4.56.

6.4. TABBING

TABBING with EDTASM files is much the same as it is with normal text files (4.30) except that SPACES ( ) are used instead of TAB characters (→). Since none of the lines in an EDTASM file will ever be justified, there was no need to use a special character other than a space to separate the words on each side of the TAB as there was with text files.

If you TAB while the cursor is over a line number, the line number will be preserved and the cursor will just move to the second space after the line number. At any other place in the line the TAB will write over the text (unless the INSERT function is on, see 4.27).

TABBING is not allowed on BASIC text files. No action will be taken if you try to implement the function when the WIDTH FIELD contains "BAS".

6.5. LINE LENGTHS

You are free to allow EDTASM and BASIC program lines to be as long as you want them to be, even exceeding one or more 64-character display lines. Some Basic interpreters, however, put an upper limit on the length of a program line that it can process. This is sometimes in the neighborhood of 240 characters. If this is true about the BASIC interpreter you are using, then please keep it in mind.

During text manipulation of EDTASM and BASIC files, ZORLOF will always assume everything from one RETURN character (←) thru the next RETURN character to be one program line. While INSERTING (4.27), all characters from the cursor thru the next RETRUN character will get shifted to the right. When DELETING (4.11), all characters from the cursor thru the next RETURN character will get shifted to the left.

6.6. PRINTER COMMANDS

Don't use PRINTER COMMANDS with BASIC or EDTASM files, simply because other programs, such as compilers and assemblers, that read these files, will not allow PRINTER COMMANDS. When you do print the file, ZORLOF's printer command defaults will take over and give you page lengths of 66 lines, 6 lines of spacing at the top, 4 lines of spacing at the bottom, and 8 characters of spacing at the left.

7. PROCESSING FORM LETTERS

A FORM LETTER, as it is spoken of here, is any text of which several copies have to be printed. Each copy will be the same as all the other copies of the run except for a couple of words or phrases here and there. The "variables" in the text might be such things as persons' names, addresses, occupations, etc. This happens most fequently with form letters, but applies just as well to legal documents, rental leases, invitations, mailing lables, and the like.

ZORLOF has the capability of processing form letters thru the implementation of two special file types: the FORM LETTER TEXT FILE and the FORM LETTER DATA FILE. These two files are described on the following pages.

To quicken your understanding of how ZORLOF processes form letters, we have included on the ZORLOF master disk a sample FORM LETTER TEXT FILE called "FORM/TXT" and a FORM LETTER DATA FILE called "FORM/DAT" to be run with it. We recommend that you look them both over, examine the format of each, and then run it for yourself on your own system. This can be done simply by loading "FORM/TXT" into memory using CLEAR-G (4.25), then hit CLEAR-P to start it. You may notice that the data file contains three data records but only two of them were used. Of the two that were used one had an ID that matched one of the characters in the FORM LETTER DATA ID QUALIFIER printer command (5.39), and the other one had no ID at all. Take out the FORM LETTER DATA ID QUALIFIER printer command in the text file and run it again. This time all three data records will be used.

7.1. FORM LETTER TEXT FILE

There is only a few things about this type of text file that are different from your everyday ZORLOF text file. The main difference is that a FORM LETTER TEXT FILE will contain the FORM LETTER DATA FILE SPECIFICATION printer command (5.38). This command tells ZORLOF two things: that this is a FORM LETTER TEXT FILE, and what the name of the FORM LETTER DATA FILE is that will be used with it. The second difference is that the FORM LETTER TEXT FILE may (it's optional) contain the FORM LETTER DATA ID QUALIFIER printer command (5.39), if only a portion of the FORM LETTER DATA FILE is to be used. The last difference is that the "variable" parts of the text will have "labels". These "labels" will be replaced by data from the FORM LETTER DATA FILE during the form letter run.

7.2. FORM LETTER DATA FILE

The FORM LETTER DATA FILE is the file that contains the list of names, addresses, phone numbers, etc, that will replace the variable labels in the FORM LETTER TEXT FILE. The FORM LETTER DATA FILE is broken into sub-units which we call data records. When the form letter is run, one copy of the FORM LETTER TEXT FILE will be printed for each data record of the FORM LETTER DATA FILE. Therefore, one data record will contain all the necessary data for one person (client, account, whatever). You may, however, choose only a select grouping of data records to be used during the form letter run, if you desire, by giving each data record an ID and then specifying which data records will be used by the FORM LETTER DATA ID QUALIFIER printer command (5.39). There is no limit to the number of data records you are allowed per FORM LETTER DATA FILE.

Each data record is also made up of sub-units, which we call data fields. The data field is one unit of information about the person (client, account, whatever). There could be one data field for his name, one for his phone number, one for his shoe size, etc. There is no limit as to the number of data fields you are allowed per data record. ZORLOF also allows data records to be any size from as small as 1 character to as large as 1000 characters. Thus, one "variable" of the form letter could be a whole paragraph or a long list of items, if you want it to be that long.

Each data field, in turn, is made up of two smaller units: the data field label and the actual data associated with that label. The label can be any string of characters (except space, RETURN character, and comma) as short as one character or as long as 28 characters. These labels should match exactly the variable labels found throughout the FORM LETTER TEXT FILE. During the time of the form letter run, all strings of characters in the FORM LETTER TEXT FILE that match data labels in a given data record of the FORM LETTER DATA FILE will be replaced by the data from the data field associated with that label. Once all the replacing is done for one data record, the modified text file is printed out.

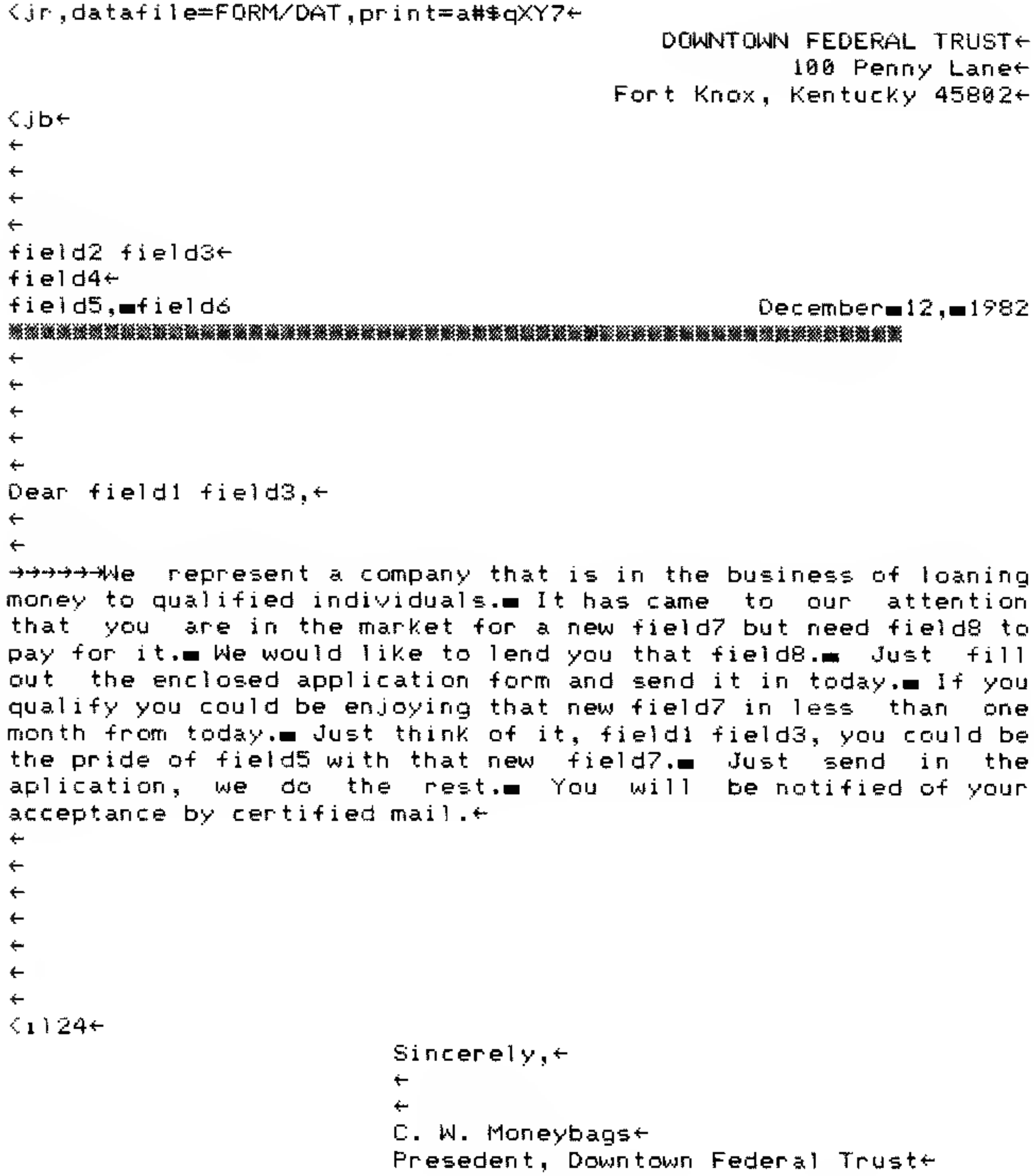
7.3. CREATING A FORM LETTER DATA FILE

Creating a FORM LETTER DATA FILE is as easy as falling off a log. First, decide whether you are going to type cast the data records allowing yourself the oppertunity to specify only records of certain types be used during the form letter run (5.39). If so, type one character (any character but space, RETURN character, or comma) for that data record's ID and hit the ENTER key. Do this at the start of every data record in the FORM LETTER DATA FILE, using a different character to indicate a different record type. If, on the other hand, you have no need to segregate the data records into group types then skip this step.

The next step is even easier. Type a one word label 1 to 28 characters long to match one of the variable labels in the FORM LETTER TEXT FILE (you make it up), type a space or a RETURN character (←), type one unit of data 1 to 1000 characters long (the guy's name or whatever), then type a RETURN character (←). The only ground rules on the data part is that it be less than 1001 characters long and contain one, and only one, RETURN character, and thats to be at the end of the data field. Repeat this for each data field of the data record.

The last step is the easiest of all. Just type one more RETURN character (←) after the RETURN character of the last data field of the data record. When ZORLOF sees these two RETURN characters in a row it will know that the data record is ended. Continue these steps, creating one data record for each copy of the form letter to be printed.

The next two pages contain one FORM LETTER TEXT FILE example and one FORM LETTER TEXT FILE example. These are the same two form letter files supplied on your ZORLOF master disk. You will notice that the data file contains three data records each having eight. The first and third data record have ID's, but the second has none. Also notice that the data field lables in the data file match the variable labels in the text file.



```
<jr,datafile=FORM/DAT,print=a#$qXY7←
                                              DOWNTOWN FEDERAL TRUST←
                                                      100 Penny Lane←
                                           Fort Knox, Kentucky 45802←
<jb←
←
←
←
←
field2 field3←
field4←
field5,▬field6                                      December▬12,▬1982
▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
←
←
←
←
←
Dear field1 field3,←
←
←
→→→→→→We  represent a company that is in the business of loaning
money to qualified individuals.▬ It has came  to  our  attention
that  you  are in the market for a new field7 but need field8 to
pay for it.▬ We would like to lend you that field8.▬  Just  fill
out  the enclosed application form and send it in today.▬ If you
qualify you could be enjoying that new field7 in less  than  one
month from today.▬ Just think of it, field1 field3, you could be
the pride of field5 with that new  field7.▬  Just  send  in  the
aplication,  we  do  the  rest.▬  You  will  be notified of your
acceptance by certified mail.←
←
←
←
←
←
←
←
<ıl24←
                        Sincerely,←
                        ←
                        ←
                        C. W. Moneybags←
                        Presedent, Downtown Federal Trust←
```

7.5. FORM LETTER DATA FILE EXAMPLE FORM/DAT

```
X←
field1 Mr.←
field2 Clark←
field3 Kent←
field4 1655 Lois Lane←
field5 Metropolis←
field6 NY▬10507←
field7 cape←
field8 $50.00←
←
field1 Miss←
field2 Sally←
field3 Somethingelse←
field4 38-24-36 Waahoo Court←
field5 Long Beach←
field6 CA▬93228←
field7 bikini←
field8 $200←
←
Q←
field1 Mrs.←
field2 Hellen←
field3 Housewife←
field4 250 Taken Place←
field5 Contentsville←
field6 IN▬59764←
field7 dish washer←
field8 $750←
←
```

A. EXPLANATION OF FILE TYPES

ZORLOF can accommodate 5 different text file types. they are: ZORLOF, Apparat EDTASM, Radio Shack EDTASM, BASIC, and ASCII. Each of these file types and the differences that make them unique are explained on the following page.

An ASCII file, as it is spoken of here, is a file, that when stored on disk, has no special codes embedded in it - just viewable text in ASCII format. This is the standard file format of most all garden variety word processors.

Because of some of the special functions that ZORLOF has, it was necessary that ZORLOF files go beyond the standard ASCII file format and add status codes at the end of each line within the text files to define that line's justification, width, and indent, so that proper line positioning would be assured as files are read from the disk for viewing.

The ASCII file format feature, present in ZORLOF, allows the operator the opportunity to save a file onto disk in standard ASCII format instead of the ZORLOF text file format if so desired. Using this capability, any file created on ZORLOF can be read by other word processing systems, and vice versa.

To save a file in ASCII format, just type the letters "ASC" in the width field and hit CLEAR-P. To save a file in ZORLOF text file format type a number (5 - 128) in the width field or leave it blank. Also any time a file is retrieved from disk, the width field will contain the file type: ASC - ASCII, BAS - BASIC, EDT - Apparat EDTASM, EDR - Radio Shack EDTASM, or a number (5-128) or blank - ZORLOF.

Note: If a file is loaded into the computer's memory for editting in ZAP format (4.58), the three letters "ZAP" will appear in the WIDTH FIELD. This does not indication a unique file type. It is instead an indication of a unique way of displaying and editing that can be done with <u>any</u> type of file.

ZORLOF TEXT FILES

The first character is a D4H, followed by three charaters that comprise the width field of the top status line, followed by the text, ending with a 00H. Each text line is followed by three line status codes, the first defines its justification, the second its width, the third its left indent. Identified by ZORLOF by the leading D4H.

APPARAT EDTASM FILES

The first charater is a D3H, followed by the first 6 characters of the file name, followed by the text, ending with a 1AH. Each line starts with 5 ASCII digits with the 8th bit set (B0H - B9H), words are separated by spaces (20H) and/or tabs (09H), and each line ends with a carriage return (0DH). Identified by ZORLOF by the leading D3H.

RADIO SHACK EDTASM FILES

This is the same as the Apparat EDTASM file except the D3H and the first 6 characters of the file name are not present in this file type. Identified by ZORLOF by a leading character of B0H - B9H.

BASIC TEXT FILES

Similar to the Radio Shack EDTASM file type except that the 8th bits of the line number digits are cleared (not set), and the ending character of the file is a 00H. Identified by ZORLOF by a leading character of 30H - 39H.

ASCII TEXT FILES

Similar to the BASIC file type except the first character of the file is any 7-bit ASCII character except a digit (30H - 39H). Identified by ZORLOF by a leading character of 21H - 2FH or 3AH - 7FH.

B. CUSTOMIZING ZORLOF

The following is a list of some customizations you can make to ZORLOF/CMD, using the ZAP-PROCESSING feature described in section 4.58. You need only make those patches which you feel apply to your situation. Chances are none of them will apply.

Be sure you understand the hexidecimal numbering system, since that is the way the addresses and the data are represented in "ZAP" format. One byte (one data value) is represented by two characters on the data line. There are 32 characters in the data portion of the line, thus there are 16 bytes represented per line. The last digit of the patch address (0-F), will always be that byte's offset within the data line. The first byte of the line will have an address that ends with "0", and the last byte of the line will have an address that ends with "F". Therefore, the byte with address 2C7E will be found on the line labeled 2C70, and it will be the 15th byte of that line (29th and 30th characters, not counting the address characters).

Once the correct byte is found for the patch, type over the old value(s) with the new values(s). When finished with all the patches you want to make, use CLEAR-F to file the new version of ZORLOF/CMD back to the disk. Be sure you only make modifications to a copy of the original program, and not to the original itself. If for any reason you feel unsure about making any of these patches, send us the original disk, a copy disk, a list of the patches you wish to have done, and $5.00 for shipping and handling, and we will do the patching for you.

B.1.

On some printers an unexpected character may appear at the start of each line. This is due to the fact that the printer cannot handle the CANCEL code (18H) that ZORLOF sends at the start of each line. This CANCEL code can be deleted using the following patch.

address	old	new
0C48	18	00

B.2.

On some older TRS-80 model 1's the cursor may seem to disappear and then later reappear. This can be remedied by slowing down the blink rate of the cursor.

address	old	new
3C1F	01	02

B.3.

If you wish to have the default for the KEYBOARD KLICK function (4.61) to be off at power up instead of on, install this patch.

address	old	new
3C7D	00	C9

B.4.

If you wish the BOLD text (5.17) on your printouts to be achieved with greater than or fewer than 8 passes over the same text, install this patch. The "#" is the number of passes to acheive BOLDED text. Epson owners ignore this patch. Your BOLDING is done thru the overstrike-emphasize print mode in the printer and does not require more than one pass.

address	old	new
0C5A	08	"#"